EXHIBIT O

NetRanger *User's Guide*

112096

NETRANGER TRAINING PRE-REQUISITES

Individuals interested in the three day NetRanger™ training course should have the following experience and/or core knowledge to maximize their learning experience:

1. TCP/IP
2. PCF - Private Control Facility
3. NetSentry
4. UNIX
5.

Produced in the United States of America.

Getting Help

WheelGroup Technical Assistance

Refer to this *User's Guide* whenever you have a problem using the NetRanger System. If you still cannot solve the problem or if you have questions about anything not covered in this guide, you can call WheelGroup Technical Support from 8:00AM to 6:00PM, CST. This assistance is available Monday through Friday (except holidays). You can reach us in any of the following ways:

Telephone	**1-888-942-6762** \ 1-000- WGC-NDOC /
FAX	**210-494-6303**
E-Mail	**help@wheelgroup.com**

Please include the following information in your FAX or e-mail message, or have it available if you are calling (Use the NetRanger Problem Report Form on the following page to help you gather this information):

- Your customer number and phone number. For e-mail messages, please include your customer number as part of the subject line for your messages.
- A description of the NSX and Director hardware and software you are using, including any network software.
  Note: To determine the version number of the Director software you are using, type the following command:
  ```
  nrdirmap -?
  ```
- Your operating system type and version
- A description of the problem, what you were doing when it occurred, and the exact wording of any error messages that you might have received.
- If this is a security question, please include all information related to a specific alarm or attacking site.

NetRanger Problem Report Form

Before you call, please have the following information available.

Organization ID:____________ Host ID (of Problem System):____________

Point of Contact:____________ Phone Number:____________

For Director or NSX Problem

NSX Information (if applicable)

NSX Type:____________ Number of NSC devices connected to NSX:________

NSC Type (with problem):____________

NSC Serial Number (with problem):____________

Director Information (if applicable)

Platform: HP SPARC

OV Version:____________ Director Version:____________

Problem Description

____

____

____

____

____

For Security Incidents

Are you a monitored site? Yes No Member of WARN? Yes No

IP addresses of involved machines:____________

Do you wish to talk to WheelGroup about
Intrusion Control and Response (ICR) consulting? Yes No

Alarm Type:____________

Description of security incident:____________

____

____

WheelGroup Corporation
1-888-942-6762
1-210-494-6303 (fax)
e-mail: help@wheelgroup.com

Table of Contents

SYM_P_0526574

5

I Overview

General Information about NetRanger's Components and Capabilities

What is NetRanger?

NetRanger is a real-time network security management system that detects, analyzes, responds to, and deters unauthorized network activity. The NetRanger architecture supports large-scale information protection via centralized monitoring and management of remote dynamic packet filtering devices that plug into networks. Communication is maintained via WheelGroup Corporation's (WGC) proprietary secure communications architecture. Network activity can also be logged for more in-depth analysis.

As shown in Figure I-1, NetRanger 1.1 includes the following core systems and subsystems:

- **Network Security eXchange (NSX)**
  - ◊ Packet Filtering Device(s)
  - ◊ Sensor
- **Communications System**
  - ◊ Post Office(s)
  - ◊ Encrypted Sleeve
- **Director**
  - ◊ Security Management Interface
  - ◊ Security Analysis Package

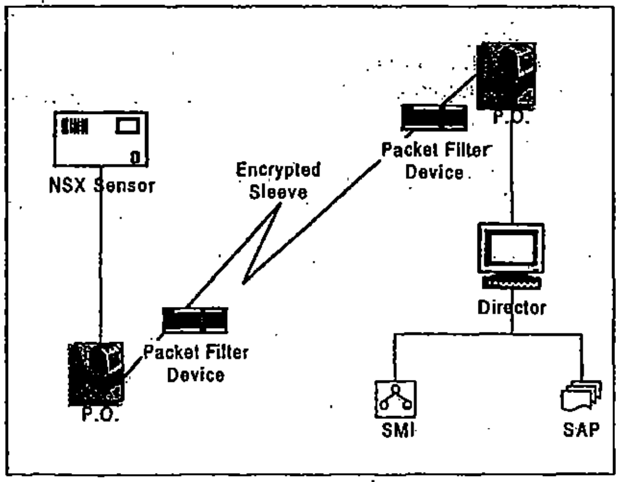


Figure I-1: Basic NetRanger Components

The NSX

The NSX is the on-site filtering, sensing, and management component of the NetRanger System. It communicates with one or more remote Director systems via the Post Office network communications system. The NSX currently interfaces directly with IP networks and supports many hardware and software configuration options.

The Packet Filtering Device is typically a router or bridge that plugs into a network at a point of entry to other networks. The security policy installed on this device determines what subset of network traffic will be routed to the NetRanger Sensor.

The Sensor subsystem contains NetRanger's real-time intrusion detection and content assessment logic. The intrusion detection engine recognizes and responds to attacks, such as sendmail, ping sweeps, IP source routing and spoofing, FTP and Telnet abuse, and SATAN scans. The Sensor's analyses produce data streams of IP packets and event records that are either dumped into local session log files or sent on to the Post Office for remote delivery to a Director system. The Sensor also accepts intrusion response and reconfiguration information from Director systems.

The Communication System

The Post Office subsystem provides remote monitoring and NSX systems management. All communication is based on a proprietary, connection-based protocol that can switch between alternate routes in order to maintain the point-to-point connections specified in its routing tables. All messages are routed based on a three-part address that includes organization, host, and application identifiers.

The Encrypted Sleeve secures data transmission between remote protected networks. Encrypted sleeves are currently implemented via the Data Privacy Facility (DPF) that comes with Network System Corporation's (NSC) BorderGuard packet filter devices.

The Director

The **Director** provides the monitoring and analysis services to NetRanger, and communicates with one or more NSX systems via the communication system. The Director contains two basic subsystems: the Security Management Interface (SMI) and the Security Analysis Package (SAP).

The SMI is a collection of GUIs and tools that help monitor and respond to security events at one or more NSX locations. The SMI integrates with network management applications (such as HP OpenView™) via menu add-ons. Whereas the SMI is focused primarily on real-time security event management, it also supports data analysis via the SAP.

The SAP is a set of data analysis tools that analyzes NSX data independently of SMI activities. The SAP consists of two basic components: database export utilities to relational databases (such as Oracle™) and one or more data analysis tools. Both types of components can be easily integrated into an existing SMI platform. However, WheelGroup Corporation recommends that all data be exported onto a separate host. In this way, the SMI and SAP components can be configured, secured, and fine-tuned independently.

Product Capabilities

The way NetRanger integrates many tried and true network security technologies makes it the preeminent intrusion detection system. Most network security applications fall into one of two categories: **firewalls** or **network monitoring.** Both of these technologies suffer from shortcomings not found in NetRanger.

Firewalls represent the most common form of network security, and they gained popularity in part because of their relatively simple host-based installation and friendly graphical user interfaces. The biggest problem with firewalls, however, is that they rely upon *proxy services* to enforce an organization's security policies. Proxy services erect *static* barriers that frequently block legitimate as well as illegitimate activity. This puts pressure on system administrators to open pathways, which makes firewalls vulnerable to the very events they are supposed to guard against. Because most firewalls are host-based solutions, administration of more than one firewall at a time is difficult. The system overhead associated with proxy services also prevents most firewalls from being able to scale beyond Ethernet speeds.

Although network monitoring tools can detect unauthorized activities without having to erect barriers to entry, administrators typically have to sift through audit logs after the fact in order to find security breaches. In many instances, systems have been compromised by the time the activity has been detected. Both of these approaches also tend to only look at incoming network traffic.

NetRanger enforces an organization's security policy via **real-time response and detection of intrusive events** without having to erect static barriers. NetRanger's secure communication architecture also allows command and control, as well as system information, to be distributed across networks. This section describes these capabilities in a top-down fashion within the context of the underlying architecture.

One of the fundamental design principles of NetRanger is that the *services* required by each of the subsystems be broken apart into their atomic operational components, or *daemons,* which are diagrammed in Figure I-2. For example, the NetRanger daemon that logs events is totally separate from the ones that perform network sensing and device management. This was done for **speed, durability, scalability,** and **independence.**

Each of NetRanger's daemon services is purpose-built for a specific task. This makes it possible to optimize each service without compromising the functionality of the other services. Purpose-built components also tend to be more durable than systems that manage multiple tasks, and are easier to debug and upgrade.

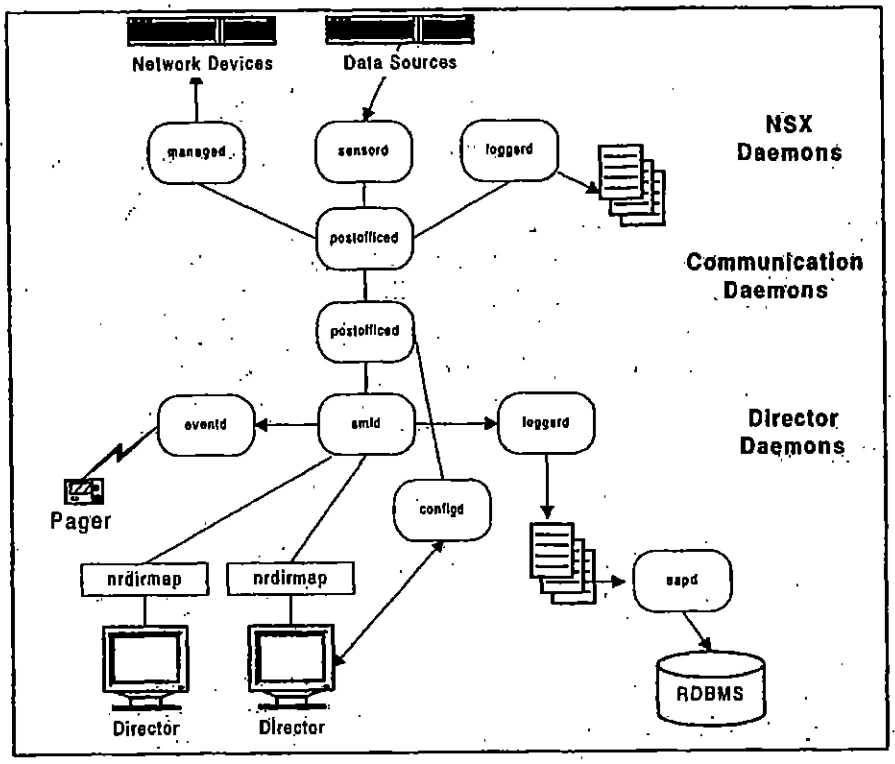


Figure I-2: NetRanger 1.1 Architecture

NSX System

The NSX's capabilities are best described as **network sensing, device management,** and **session logging,** implemented respectively by the *sensord, managed,* and *loggerd* daemons diagrammed in Figure I-2.

Although NetRanger is frequently described as an *Intrusion Detection* system, it also looks for a variety of suspicious activities that precede unauthorized events. An NSX will detect and report a ping sweep of a network. Although not truly intrusive, a ping sweep is frequently a precursor to unauthorized activity. The NSX system therefore looks for network **patterns of misuse** based on a variety of different **attack signatures.**

Network Sensing

Patterns of misuse are identified by two basic types of network signatures: **context** and **content.** Context-based signatures deal with the *state* of a transmission as defined by the structure of packet headers, and content-based signatures focus on *what* is being transported—the binary data. Context-based signatures tend to be more complex than content-based ones and the steps required to identify them are also complex and proprietary. Consequently, context-based and content-based signatures are embedded within the NSX sensor subsystem. In addition, simple content-based signatures can be added dynamically at runtime.

Network Protocols

The sensor subsystem currently works with **TCP/IP**. Future releases of NetRanger will support other network protocols, such as Novell's IPX/SPX.

Packet Filter Devices

The only type of packet filter devices the sensor subsystem currently works with are the **BorderGuard** and **ERS/Passport** devices from Network Systems Corporation (NSC). These packet filter devices play a key role in the success of the NSX system.

In addition to serving as high-speed IP data sources, all of these devices

- can be **reconfigured on the fly,**
- support the same **NetSentry interface,**
- can be deployed as **bridges** as well as **routers**, and
- can maintain **Virtual Private Network** (VPN) connections.

Because these devices can be reconfigured on the fly, NetRanger can dynamically **shun as well as detect** suspicious and unauthorized network activity. The common command and control interface provided by NetSentry allows one NSX to support all three devices. It will soon be possible to operate these devices as bridges or routers, which means that an NSX can be deployed in a network behind existing devices, such as Cisco routers, without having to change routing protocols or reassign existing network addresses. The VPN facilities provided by NSC are the key to NetRanger's secure communication system.

Performance Capabilities

The NSC packet filter devices fall into three distinct price/performance products: the BorderGuard 1000, BorderGuard 2000, and the ERS/Passport. These packet filter devices serve as the basis for the three NSX configurations currently offered by WheelGroup: the NSX 1000, 2000, and 5000 systems. As with the NSC systems, the primary difference between the NSX systems is one of network performance, which is summarized in Table I-1. For detailed hardware information on any of these systems, refer to Appendix D in this *User's Guide*.

	NSX 1000	NSX 2000	NSX 5000
Max Bandwidth	512 Kbps	T1 (or 10 Mbps)	T3 (or 100 Mbps)
NSC Device	BorderGuard 10000 > 2 Ethernet > 3 WAN/1 Ethernet	BorderGuard 2000 > 4 Ethernet > 2 Ethernet/2 WAN	ERS/Passport > FDDI > Ethernet > WAN > Token Flag
NSX Sensor	Pentium 166 Mhz > 2 GB Hard Drive > 32 MB RAM > 10BaseT Ethernet > modem dial-up > 2 serial ports	Pentium 166 Mhz > 2 GB Hard Drive > 32 MB RAM > 10BaseT Ethernet > modem dial-up > 2 serial ports	UltraSPARC 143 Mhz > 4 GB Hard Drive > 64 MB RAM (min) > SBUS FDDI > modem dial-up > 2 serial ports

Table I-1: NSX Configurations

NetSentry

As noted earlier, NSX intrusion detection is based on the **monitoring of an open network connection rather than a closed one**. The NSX does this by leveraging the layered filter architecture built into NetSentry, which is diagrammed in Figure I-3.

The *First*, *Apply Table*, and *Last* filter points apply to **all** of the network interfaces installed on a BorderGuard/ERS system, whereas the *Incoming* and *Outgoing* filter points allow different **in** and **out** policies to be applied to each of the network interfaces installed on the device.

One of the reasons the NSX system is able to perform **real-time** intrusion detection is because it is able to leverage the **copy_to/log_to** auditing features via filters applied to the *First* filter point. For example, the NSX's default *first.fil* filter instructs the BorderGuard/ERS to only pass on specific IP packets to *sensord*. This helps to dramatically reduce the amount of traffic the NSX system has to process.

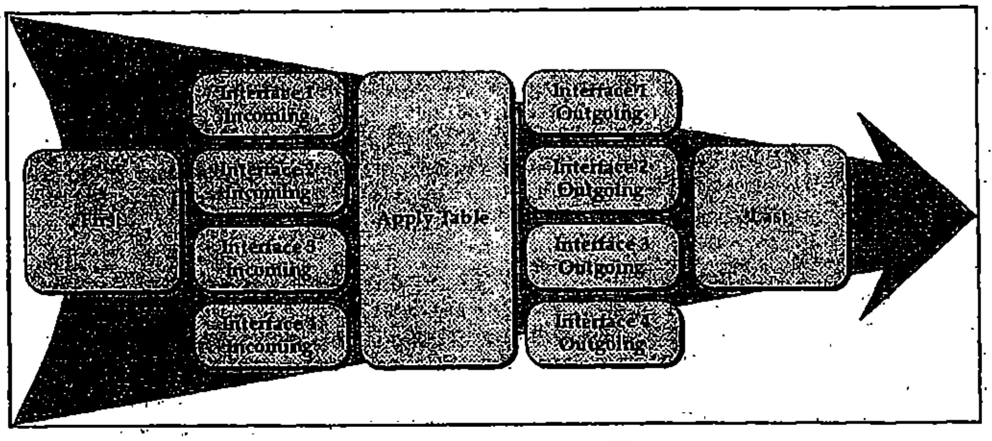


Figure I-3: NetSentry PCF Filters

Attack Signatures

As previously mentioned, an attack signature is a *pattern of misuse* based on one or more events. Such a pattern can be as simple as an attempt to access a specific port on a specific host, or as complex as sequences of operations distributed across multiple hosts over an arbitrary period of time. Events can be grouped into different *attack signatures*. An event that is based on a single ICMP packet at a specific point in time is an **atomic signature** (e.g., a ping of a specific host). **Composite signatures**, on the other hand, are based on series of events. A ping sweep is an example of a signature that spans a network; a port sweep is an example of a signature that focuses on a specific host. A SATAN attack is an example of a composite signature derived from a host port sweep pattern. Many of these patterns are also *stateless*, which means that the attack signature must be identified regardless of the order or the duration between atomic events. Other signatures, such as SYN attacks, are based on well-defined event sequences that are *stateful*.

Attack Responses

The NSX system does one or more of the following things once an attack has been positively identified:

- **Generate an alarm**
- **Shun the attack**
- **Log the alarm event**

In keeping with the flexibility of NetRanger, the initiation of these actions, as well as how they are initiated, is highly configurable.

Alarms

Alarms are generated by the *sensord* daemon, and are typically routed to a remote Director system. These notifications can also be routed to *multiple* Director systems.

The type of alarm generated for a specific attack is dictated by configuration tokens stored in the NSX configuration file /usr/nr/etc/sensord.conf. Alarm notifications are grouped according to their **severity of misuse.** NetRanger allows you to define up to 255 different levels of severity. However, the default configuration currently specifies 6 levels of misuse.

Shunning

In addition to generating alarms, *sensord* can initiate **optional actions.** The most common action is to **shun an attack,** which typically involves reconfiguration and reloading of the NetSentry *Apply Table* (as diagrammed in Figure I-3). This type of automated response should only be configured for attack signatures with a low probability of a *false positive* response. A SATAN attack is an example of an unambiguous activity, whereas a content-based signature such as "vrfy" (off mail port 25) is more prone to a false positive pattern match.

Another way of shunning patterns of misuse is to manually reconfigure the BorderGuard or ERS device through the Director system. Shunning is part of a site's security policy that must be carefully reviewed before it is deployed, whether as a set of automatic rules in sensord.conf, or as a set of guidelines that operational staff rely upon.

Logging

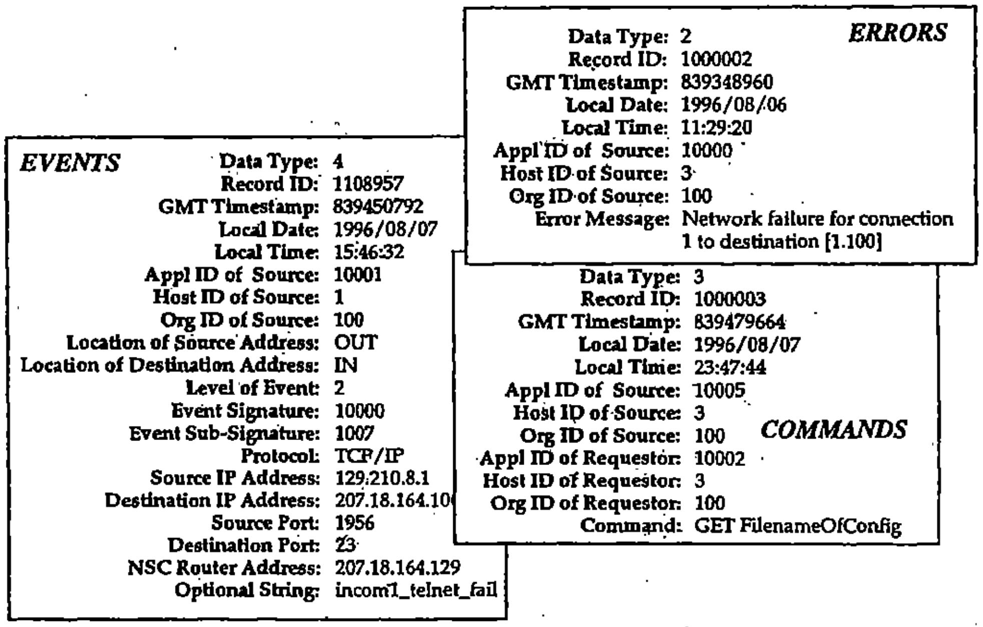
All NSX log data is written to **flat files,** which can then be exported to industrial-grade databases by the SAP subsystem described in the *Director* section. Data is written directly to flat files instead of a database in order to maximize **fault tolerance** and **performance.**

The NSX system currently supports two types of logging:

- **Event logs**
- **IP Session logs**

Alarms represent just one type of *event* that can be logged by the NSX system. Event logs can also contain entries for every *command* and *error* that is generated by a user or daemon service. Data is written to these type of log files as long as NetRanger is running. Log files are **serialized** based on configurable time and size intervals defined in /usr/nr/etc/loggerd.conf. The naming convention for Event log files is **log.<date-time>.**

IP Session logs are only written to when a certain event(s) occurs, such as a connection request from a specific IP address, or detection of a string such as "Confidential." When these type of conditions are met, *sensord* can be configured to write every incoming and outgoing packet to an IP Session log for a predefined period of time. IP Session logs allow you to reconstruct the *conversation* that takes place between a source and destination IP addresses. The naming convention for IP Session logs is **log.<src IP address>.**



ERRORS

Data Type:	2
Record ID:	1000002
GMT Timestamp:	839348960
Local Date:	1996/08/06
Local Time:	11:29:20
Appl ID of Source:	10000
Host ID of Source:	3
Org ID of Source:	100
Error Message:	Network failure for connection 1 to destination [1.100]

EVENTS

Data Type:	4
Record ID:	1108957
GMT Timestamp:	839450792
Local Date:	1996/08/07
Local Time:	15:46:32
Appl ID of Source:	10001
Host ID of Source:	1
Org ID of Source:	100
Location of Source Address:	OUT
Location of Destination Address:	IN
Level of Event:	2
Event Signature:	10000
Event Sub-Signature:	1007
Protocol:	TCP/IP
Source IP Address:	129.210.8.1
Destination IP Address:	207.18.164.10
Source Port:	1956
Destination Port:	23
NSC Router Address:	207.18.164.129
Optional String:	incom1_telnet_fail

COMMANDS

Data Type:	3
Record ID:	1000003
GMT Timestamp:	839479664
Local Date:	1996/08/07
Local Time:	23:47:44
Appl ID of Source:	10005
Host ID of Source:	3
Org ID of Source:	100
Appl ID of Requestor:	10002
Host ID of Requestor:	3
Org ID of Requestor:	100
Command:	GET FilenameOfConfig

Figure I-4: Event Log Formats

Timestamp	Packet Length	IP Packet
4 bytes	4 bytes	20+ bytes

Figure I-5: IP Session Log Format

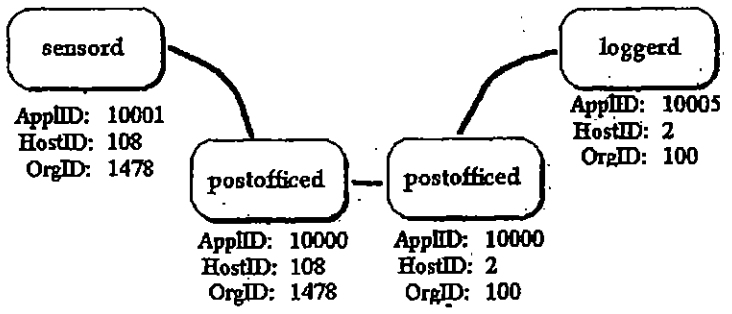
Note that both Event and IP Session information can be logged locally on the NSX system and remotely on Director systems; exactly **what** information is sent **where** depends on how each NSX system is configured.

Communications System

NetRanger is a **distributed application** that allows exchange of audit information and command and control across networks. All communication is based on a **proprietary, connection-based protocol** that is **fault tolerant** and supports **alternate routes.** Although the communication system currently only runs on top of TCP/IP networks, it has been implemented at the **application layer** of the OSI network model to eventually operate on top of other protocols, such as IPX/SPX and NetBios. The underlying packet transfer mechanism employs a **sliding window** for performance and is **UDP-based** for scalability.

Communications Protocol

As Figure I-6 illustrates, all of the NetRanger daemons communicate with one another via *postofficed* daemons. Note that this holds true for communication between daemons on the same host as well as across hosts. All communication is based on a unique three-part address that includes **Organization, Host,** and **Application** identifiers, which are enumerated in each NSX's and Director's *organizations, hosts,* and *services* files in /usr/nr/etc.



100	wheelgroup
1000	company Aaaa
1478	company Aaab
3000	company Abbb

organizations

4.100	localhost
1.100	netranger2.wheelgroup
2.100	marauder.wheelgroup
3.100	firefly.wheelgroup
4.100	sentinel.wheelgroup

hosts

10000	postofficed
10001	sensord
10002	configd
10003	managed
10004	eventd
10005	loggerd
10006	smid
10007	sapd

services

Figure I-6: Example /usr/nr/etc communication files

The benefits of this proprietary addressing scheme are twofold: 1) it can be layered on top of existing network protocols and 2) it can address a much larger domain than the current 32-bit IP protocol.

Alternate Routes

NetRanger's three-part addressing scheme serves as the basis for a point-to-point protocol that allows for up to 255 alternate routes between two hosts. These alternate pathways are defined in /usr/nr/etc/routes, which also maps NetRanger addresses to native host addresses. Figure I-7 shows two alternate routes to the "wheelgroup" host "netranger2". The preferred path is via an IP host with the address of 207.18.164.130; an alternate path has been identified via a host with an IP address of 10.1.4.150.

```
netranger2.wheelgroup  1  207.18.164.130    45000  1
netranger2.wheelgroup  2  10.1.4.150        45000  1
marauder.wheelgroup    1  10.1.4.150        45000  1
```

Figure I-7: Example /usr/nr/etc/routes file

An important feature of this alternate routing protocol is that it automatically switches to the next route whenever the current route fails. It also uses a system heartbeat to detect when a connection to the preferred route can be reestablished. A system error message is generated (and logged) whenever a connection goes down, and any packets that were lost during a state transition are resent.

Distribution of Data

In addition to specifying alternate routes and maintaining fault tolerance, the communication protocol also allows you to define **what** types of information and **how much** information should be sent to each destination.

As noted earlier, the types of information generated by the NSX daemon falls into four basic categories: **Events, Commands, Errors,** and **IP Packets.** The /usr/nr/etc/destinations file allows you to specify what types of information should be routed to which **daemons** on which **hosts.** Figure I-8 shows two different distribution entries. The first entry routes all of the standard Event data to the *loggerd* service on a Director platform named *sentinel*, and the second entry specifies that only *events* should be displayed by *smid* on *firefly* Director platform.

```
1 sentinel.wheelgroup    loggerd  1  EVENTS,ERRORS,COMMANDS
2 firefly.wheelgroup     smid     2  EVENTS
```

Figure I-8: Example /usr/nr/etc/destinations file

In addition to specifying what *types* of information should be distributed, NetRanger can dictate what *levels* of event should be sent to each destination. As shown in Figure I-8, only events of level "2" *and above* are being sent to *smid* on *firefly.*

Distribution Hierarchies

Another feature that complements alternate routing is the ability to build hierarchies of NSX and Director systems through the use of **message propagation.** Instead of broadcasting events from an NSX onto multiple hosts, information can be sent to a single host, which can then propagate packets onto other platforms defined in its local configuration files. Figure I-9 illustrates this concept via a simple hierarchy of Director platforms.

In addition to providing a degree of fault tolerance, distribution hierarchies can simplify system management. For example, local Director Platforms might be responsible for monitoring from 9 am to 5 pm and then transfer control onto a central Director every evening.

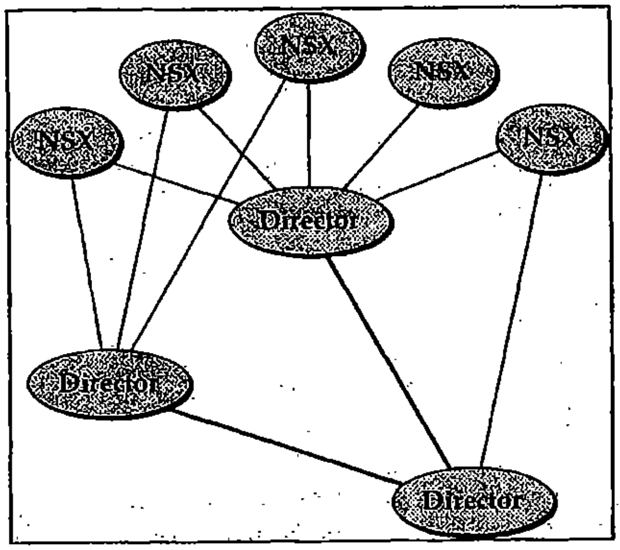


Figure I-9: Director Hierarchy Based on Message Propagation

Encrypted Sleeves

A key requirement of the communication architecture is its **security,** which is achieved by passing all communication between NSX and Director systems through NSC's DPF. The DPF maintains Virtual Private Networks **(VPN)** via **RSA** public key and one of the following private key systems: **DES, Triple DES,** or **IDEA**. The DPF is noteworthy because

- it can be **maintained across collections** of NSX and Director systems, and
- it is **transparent** to applications such as NetRanger.

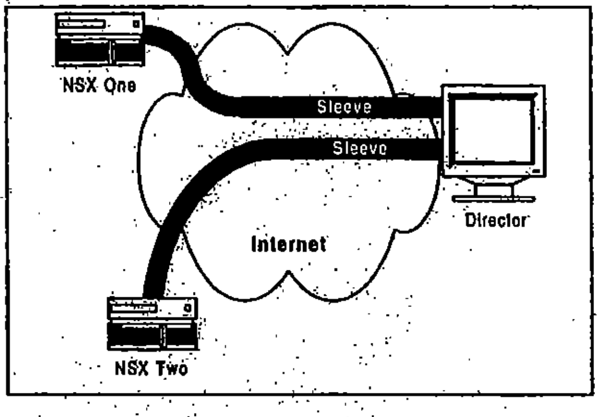


Figure I-10: Example of NSC's VPN

Director System

As noted earlier, the NetRanger Director consists of two major subsystems:

- **Security Management Interface (SMI)**
- **Security Analysis Package (SAP)**

These two subsystems provide **centralized command and control** of an organization's security perimeter, which could conceivably encompass hundreds of NSX systems. From a capabilities perspective, these two subsystems provide **monitoring, management, data collection, data analysis,** and **user-defined actions** services. All of these capabilities are supported by the *smid, configd, loggerd, sapd,* and *eventd* daemons diagrammed in Figure I-2. There is also an application called *nrdirmap* that serves as the interface between *smid* and HP OpenView.

NSX Monitoring

The Director's most prominent capability is display of **real-time** event information, which is based on the *smid* daemon, the *nrdirmap* application, and OpenView. The *smid* daemon accepts incoming event records from one or more NSX systems via a local *postofficed* and translates them into a format that *nrdirmap* understands. The *nrdirmap* application uses the OVW API (OpenView Windows Application Programming Interface) to tell the OpenView user interface what security information to present to the user. Security information is presented via **icons** drawn on one or more **network security maps.** Figure I-11 shows an example of the Director's display of a collection of NSX systems.

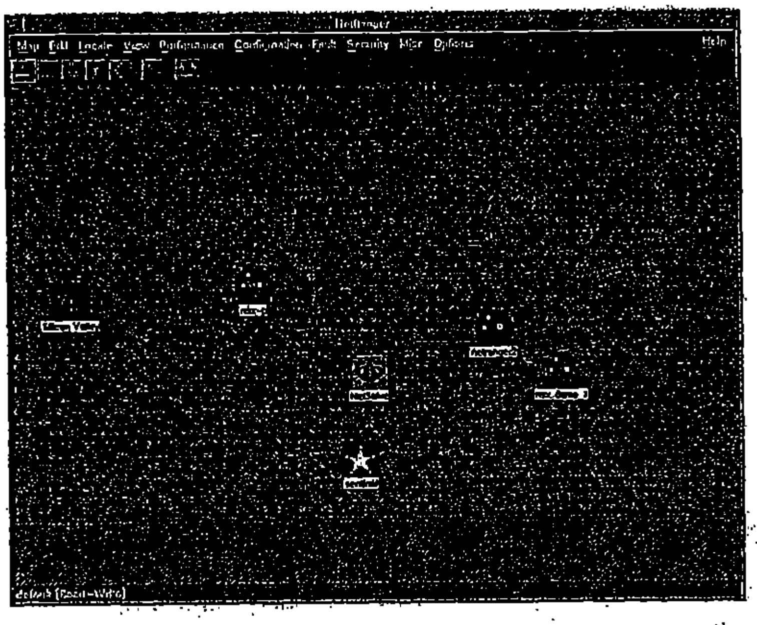

NetRanger
Map Edit Locate View Performance Configuration Fault Security Misc Options
Help
default [Read-Write]

Figure I-11: Example Director Network Security Submap

Network Security Maps

The Director arranges icons into hierarchical security maps based on OpenView's Network Node Management (NNM) user interface. If you double-click on an icon, you can view the next lower "submap." Figure I-12 shows several examples of *Event/Alarm* icons. The Director hierarchy includes the following levels, where *NetRanger* is the "root" and *Events/Alarms* are the "leaf" nodes of the tree:

- **NetRanger**
  - **Collections** of Directors/NSXs
    - A Single **Director or NSX** System
      - **Applications** running on a Director or NSX System
        - **Events/Alarms** generated by an Application

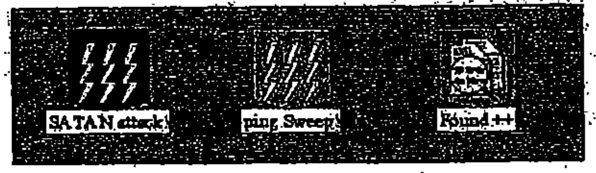


Figure I-12: Sample Director Alarm Icons

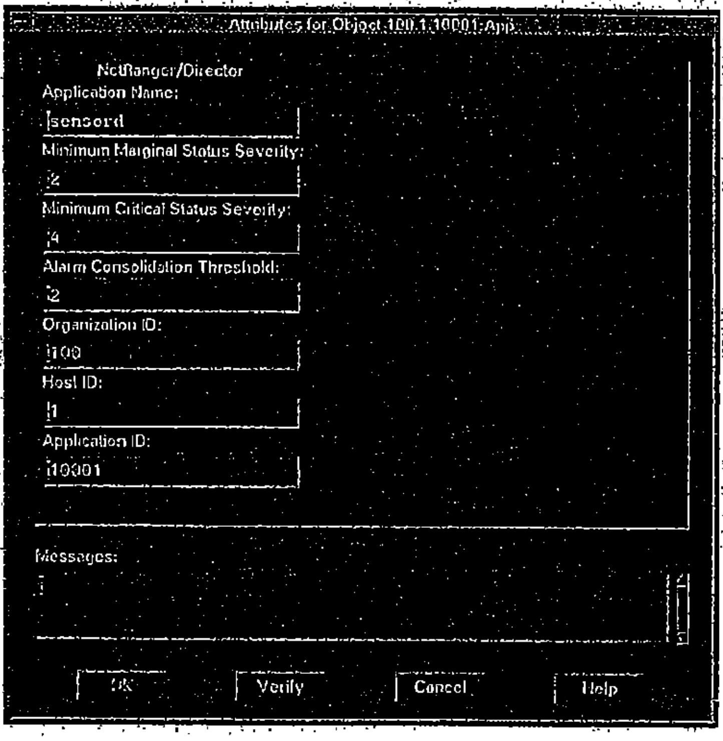
Icon States

Every icon within a network security map has a **state**, which is expressed in the form of textual and graphical attributes.

The most visible indication of an icon's state is its **color.** The colors **green, yellow,** and **red** represent the states **normal, marginal,** and **critical**. These states are user-defined in an attribute dialog (shown in Figure I-13); editable fields appear in gray. Level 2 and 3 alarms are usually set as *marginal* (yellow), and level 4 and 5 alarms are set as *critical* (red).

Unlike all other icon attributes, color is a state that is **inherited** from *Event/Alarm* icons by all other levels of the icon hierarchy. For example, an NSX *collection* icon is automatically set to red if any of its subordinate NSX applications has received a critical alarm. The ability to **propagate** alarm states up through a hierarchy is one of the features that makes the Director such a powerful network security monitoring tool.

Every icon in a Director hierarchy also possesses textual attributes. This information is accessed by selecting a particular icon and then choosing **Describe/Modify** from the menu bar or pressing **Ctrl+O,** which brings the icon's attributes dialog forward. Each type of icon has a different set of attributes. Figure I-13 shows the attributes for a *sensord* icon.



NetRanger/Director

Application Name: sensord

Minimum Marginal Status Severity: 2

Minimum Critical Status Severity: 4

Alarm Consolidation Threshold: 2

Organization ID: 100

Host ID: 1

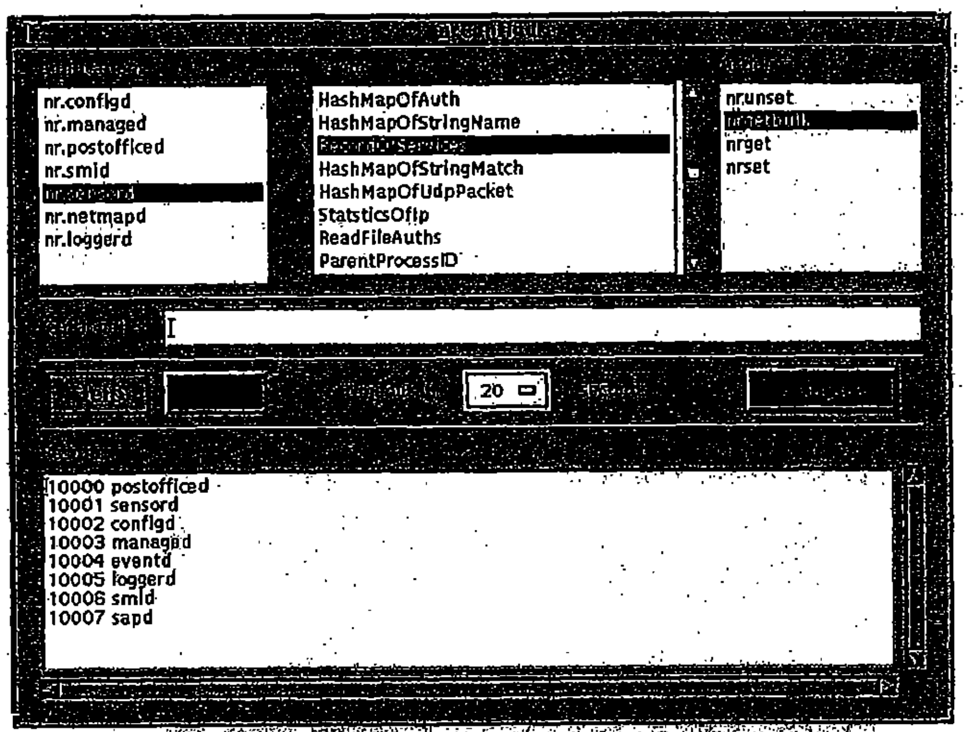
Application ID: 10001

Messages:

Figure I-13: sensord Attribute Information

NSX Management

Another key capability of the Director is remote management of the daemon processes (or *Applications*) that make up an NSX system. *Applications* are managed via a graphical user interface called **nrConfigure,** which talks directly to the *configd* daemon. This interface allows you to effectively *get* and *set* such attributes as log file names and alarm responses. The nrConfigure interface is activated by selecting one or more icons on the security map and then choosing **Security >Configure** from the OpenView menu. This opens the window shown in Figure I-14.



nr.configd
nr.managed
nr.postofficed
nr.smid
nr.netmapd
nr.loggerd

HashMapOfAuth
HashMapOfStringName
HashMapOfStringMatch
HashMapOfUdpPacket
StatisticsOfIp
ReadFileAuths
ParentProcessID

nrunset
nrget
nrset

```
10000 postofficed
10001 sensord
10002 configd
10003 managed
10004 eventd
10005 loggerd
10006 smid
10007 sapd
```

Figure I-14: NSX Configuration Interface

The **Applications** displayed in nrConfigure's top left-hand list box represent the services running on the first NSX icon highlighted by the user. The top center list box displays all of the **Tokens** that apply to the Application currently highlighted in the left-hand list box. The top right-hand list box displays the **Actions** that are allowed for the currently selected Token. Figure I-14 shows the Actions and Tokens for the *sensord* daemon.

NetRanger currently supports over 100 tokens. An *Action* is put into effect by pressing the **Execute** button. The five possible actions are **get, getbulk, set, unset,** and **exec.**

The **get** and **getbulk** commands are read-only operations that obtain information from an *Application*. The **get** command returns information for a *single* item; **getbulk** returns information for a *set* of items. Figure I-14 shows the results from a getbulk request for all of the services defined for an NSX or Director.

The **set** command is a write operation that updates the value of a *Token*; **unset** removes a specific Token and its value from an Application's configuration. Adding or deleting a content-based attack signature (such as "vrfy") from an NSX is one example of these *Actions*.

The **exec** command applies to those *Tokens* that can instruct an application to execute an action external to itself. For example, you can instruct an NSX to shun a specific IP address by issuing an **exec** against *managed*, which in turn modifies the filter configuration on its packet filter. The exec command is also used for such tasks as writing configuration information to disk.

It is important to understand that all configuration data is read and written to or from the *Applications* themselves, not from local caches or parameter files. The results of a **set** or **exec** take effect immediately on the target NSX.

Finally, all of the functionality provided by nrConfigure is also available via command-line interfaces. This allows trusted users without access to a Director to manage NSX systems from a simple terminal session.

NSX Data Collection

NSX data collection serves as the foundation for the SAP subsystem, and is based on the *loggerd and sapd* services diagrammed in Figure I-15. These daemons use a simple **push-pull** mechanism to migrate data into a remote database. As explained earlier, *loggerd* pushes data into flat files, which are serialized based on a configurable size or time interval. This data is then pulled into a remote database by *sapd*, which has its own polling interval.

Writing to intermediate flat files in this manner provides levels of **fault tolerance** and **performance** unachievable when writing directly to a database. Data throughput in a distributed application such as NetRanger is constrained by the weakest link in the system.

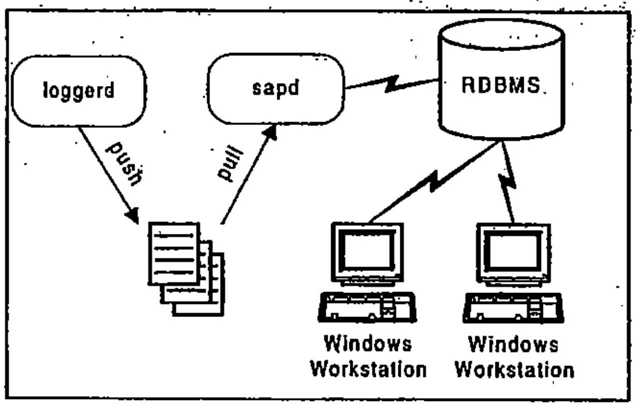


Figure I-15: NSX Data Collection

With the SAP system, the data capture process is insensitive to database availability or performance fluctuations.

The SAP currently ships with *sapd* drivers for **Oracle** and **Remedy.** However, SAP can also be configured to write to other databases, such as Sybase and Informix. Example scripts are shipped with the Director that show how a database's native bulk load tools can be easily integrated into *sapd*.

NSX Data Analysis

The second SAP capability is data analysis. Rather than locking the user into a single tool on the Director platform, it is assumed that this task is better served by **third-party tools** on a separate Windows platform, such as the IQ Objects **report writer** from IQ Software and **multi-dimension analysis** tools such as PowerPlay from Cognos. **Trouble ticketing** systems such as Remedy's Action Request System (ARS) can also be implemented on top of NetRanger's alarm data.

These type of third-party tools can be configured to support ad hoc queries as well as predefined reports. For example, these tools can easily generate reports showing

- all alarms of levels 4 and 5 in the last 30 days,
- a graph of Web server activity over the last 24 hours, and
- a table of all events in the last 30 days in order of increasing alarms.

User-Defined Actions

In addition to displaying and logging alarm events, the Director can generate user-defined actions via *eventd*. A typical action might be to generate pager notifications via e-mail, or feed data onto third-party devices, such as a printer. Support for multiple action scripts is also provided. Also, *eventd* makes no distinction between alarm types and levels. However, the default action script shipped with this service shows how to trigger actions based on these criteria.

NetRanger 1.1 Enhancements

This section identifies all of the major changes and enhancements associated with the 1.1 release of NetRanger. This information is presented within the context of the three main NetRanger systems: NSX, Communications, and Director.

NSX

Regular Expression Parsing

Content Signatures are now based on Regular Expressions rather than literal strings. In the 1.0 version of the NSX, sensord.conf contained multiple entries for a given string in order to account for such things as differences in case, leading and trailing spaces, and so on. The 1.1 version of the NSX is able to account for all of these variables through the use of regular expression syntax. For example, all combinations of "vrfy" attacks against the sendmail port can now be expressed simply as [Vv][Rr][Ff][Yy].

New Signatures

The following new attack signatures have been added into the NSX system: RPCs, DNS, WEB, YP, improved sendmail, TFTP, SYN denial-of-service, and TCP hijacking.

Installation and Configuration

The 1.1 version of NetRanger can be installed on top of a 1.0 release with minimal disruption of existing configuration information. The only files you may need to merge are sensord.conf and signatures in the ~/etc directory and *first.fil* and *incom1.fil* on the BorderGuard. The configuration of a new NetRanger's routing and filter information has also been greatly simplified. These tasks are now performed via a new process called **nsxinstall**, which resides in /usr/nr/lib.

Communications

Sliding Window

NetRanger now uses a *sliding window* to send packets between post offices. This both improves both performance and reduces the number of packets that are dropped and must be retransmitted.

Automatic Protocol Version Identification

In order to be backward-compatible with prior versions of NetRanger, the 1.1 release communicates with 1.0 and 1.1 NSX and Director systems.

Failure Under Heavy Loads

Internal tests have shown that the 1.0 version of the communication system can fail with a *core dump* when subjected to heavy load. This has been fixed on the 1.1 release.

Multiple and Read-Only Maps

With the 1.0 version of the Director, only one instance of the user interface could run concurrently. Version 1.1 allows multiple instances of the network management interface to run concurrently on the Director platform. An important benefit of this enhancement is that users at different locations can simultaneously view a common set of network security information.

Ease of Use Features

The following features have been added to the Director system to improve usability:

- The Alarm window displays its NSX machine name.
- The Director machine icon can be deleted and recreated without disrupting OpenView.
- Machines that are manually added automatically have their subordinate application icons displayed.
- Application names are automatically assigned to icons that are dragged onto a map.
- The *nrdirmap* application now understands the /usr/nr/etc/organizations file.
- The icon labels for security violation alarms are based on NetSentry filter names. In version 1.0, all security violations had the label "Sec Violation." Each icon now contains the name of the filter that detected the pattern of misuse.

New Menu Options

The following menu options have been added to the Director interface:

- **Security->About** invokes an About Box that identifies the version of *nrdirmap*.
- **Security->Show Names** identifies the *source* and *destination* of an alarm event via *nslookup*.
- **Security->Save To File** dumps the contents of the OVwDB database to an ASCII file. This information can then be attached to an email message or embedded in a document.
- **Security->Configure** invokes the new Java-based **nrConfigure** interface, which is described on the following page.

Event Icon Consolidation

In the 1.0 version of the Director, the *nrdirmap* application displayed an icon for every alarm event it received. This quickly led to a cluttering of both the screen displays and the underlying OpenView object database. This is especially true when an NSX detects recurring events, such as ping sweeps from a specific IP address.

Version 1.1 of the Director contains the ability to consolidate duplicate alarms into a single icon and OpenView database entry. A counter on the icon indicates when a subsequent alarm of that type has been received. The threshold for consolidating duplicate alarms into a single icon/entry is configurable.

Database Staging

The Director system can now stage event data from its flat files onto Oracle and Remedy databases via the *sapd* application. This capability is based on a simple *push-pull* model, in which data is pushed into flat files by *loggerd* and pulled into a remote database by *sapd*. This data can also be staged into other databases that possess native bulk loader tools, such as Informix or Sybase. Users can refer to the bulk load control files for Oracle SQL*LDR that are shipped with the *sapd* package for integration of other databases into the *sapd* package. This staging is highly configurable.

Data Analysis

NetRanger 1.1 performs data analysis of event data via third-party Windows tools by virtue of its Database Staging capability described above.

Remote Configuration

Users can now configure both local and remote NetRanger daemon applications via a Java-based configuration interface called **nrConfigure.** This interface is accessed from the Director via the **Security->Configure** menu option.

User-Defined Actions

In addition to displaying and logging alarm events, the 1.1 version of the Director can now generate user-defined actions via the *eventd* application. For example, *eventd* can generate pager notifications via e-mail for alarm events above a configurable threshold (e.g., level "4" alarms).

II NetRanger Pre-Installation

What to do Before Installing NetRanger 1.1

Before installing NetRanger, you should have a complete understanding of the current architecture of your corporate network(s). This is a critical step in the pre-installation process. Failure to obtain proper information and proceeding with NetRanger installation could result in network security holes and loss of network functionality and services. This section describes the steps you should take prior to installing the NetRanger system on your network (both the BorderGuard and the NSX).

Analyze Your Current Network Architecture

The first step in protecting your network is understanding that network's existing architecture and requirements. The most effective way to analyze your current network architecture is to follow these steps:

1. ***Identify what you want to protect.***
2. ***Define all entry and exit points to the protected network.***
3. ***Identify current security measures.***

Identify What You Want to Protect

Think about which assets on your network need protection. Also, closely examine the connectivity *between* corporate networks that might give unwanted access to your network environment. If your configuration contains connections between more than one physical site, determine whether you will protect the remote assets. If not, then determine whether you need to protect your network *from* the remote site.

Define All Entry and Exit Points to the Protected Network

This is the most important, and possibly the most difficult, step in protecting your network. The way your organization is connected physically as well as electronically will affect how you identify all network entry and exit points. Some typical scenarios you may encounter during the identification process follow.

Case 1—One Geographic Location, No Existing Internet Connection

This is the simplest case. If you are installing NetRanger in conjunction with obtaining Internet connectivity, you can install the NetRanger NSX as your Internet router and intrusion detection and response system. If you want to further isolate a single internal network from all other corporate networks as well as the Internet, identify all routers local to that network and either replace them with the NetRanger NSX or place the NetRanger NSX behind them. Refer to the *BorderGuard Setup* and *BorderGuard Configuration* sections in this *Guide* for the specific procedures you would employ in this situation.

If your network connects to the network(s) of any business partners, your organizational security posture will be enhanced by placing the NetRanger NSX on that connection, because you have no control over how a business partner's network is configured or what security countermeasures they have put in place to deter threats.

Case 2—One Geographic Location, Existing Internet Connection

Your organization has only one physical site, but more than one Internet connection. You must identify each Internet connection for NetRanger to provide the highest level of security for your network(s). To do this, contact your Internet provider to verify your connectivity and registered Internet address range. Each registered address range assigned to your company should be accounted for in your existing Internet router.

If you want to further isolate an internal network from all other networks (corporate as well as the Internet), identify all routers local to that network and either replace them with a NetRanger NSX or place a NetRanger NSX behind them. Refer to the *BorderGuard Setup* and *BorderGuard Configuration* sections in this *Guide* for the specific procedures you would employ in this situation.

If your network connects to the network(s) of any business partners, you should use a NetRanger NSX between that connection, especially because you have no control over how a business partner's network is configured.

Case 3—Multiple Geographic Locations, Existing Internet Connection

This is the most complicated situation. It is very likely that each site may have its own Internet connection as well as a path to the network you want to protect. In this case, you have three options for securing your network(s):

- **Protect the network at the Internet connection and protect it at the connection(s) to other sites**—This will secure all entry and exit points to your protected network. However, the remote sites will remain unprotected and your protected network will be more directly accessible to potential intruders.

- **Protect your local network and all remote network sites at their connections to the Internet**—This protects your corporate network from outside intruders, but it does not regulate communication between your corporate network and your remote sites or the network(s) of your company's business partners.

- **Place NetRanger on your local network, at each remote site, at all connections to the Internet, and between your corporate connections to remote sites and those of your company's business partners**—This provides the highest level of security and protects your corporate network from unwanted attack(s).

You should closely coordinate with the network administrators at your remote sites to ensure complete coverage for either of the last two options.

Identify Current Security Measures

How Existing Firewalls Affect NetRanger Installation

The NetRanger NSX can work in conjunction with existing firewalls as long as they are appropriately placed within the network. Ideally, the NetRanger NSX should be located in front of the existing firewall to provide maximum intrusion detection and response functionality. If the NetRanger NSX packet filter is placed behind a firewall, intrusion detection will be limited to traffic that has been allowed through the firewall. Therefore, it will not generate alarms on reconnaissance or possible failed malicious activity. In many cases, NetRanger can replace firewalls. If there is no need for address translation or other proxy services, NetRanger will work best without a firewall and will have less impact on network performance.

How Security Filters in Existing Routers Impact NetRanger Installation

Please note that filters on existing routers must be removed for NetRanger to function **at its optimum level.** NetRanger will provide the same filtering capability in addition to its robust intrusion detection and response capabilities.

NSC BorderGuard Installation Options

After you have analyzed your current network architecture and its existing configurations, you must then decide where to place the BorderGuard router on the network. This section describes the various placement options for the BorderGuard to ensure the highest level of security for your corporate network(s). For specific information about the configuration files for installing the BorderGuard, please refer to the *BorderGuard Configuration* section in this *Guide*.

Option 1—Install the BorderGuard as the Internet Router

The simplest way to install a BorderGuard on your network is as an Internet router. A BorderGuard 1000 or 2000 is usually connected to a Local Area Network (LAN) via an Ethernet port(s). Connections to the CSU/DSU depends on the type of unit you have. Most CSU/DSU units have a V.35 serial connection; a small percentage use RS232 or RS449.

Your company's Internet Service Provider (ISP) will assign the external IP address; you will need to provide the internal IP address(es). This option is recommended only if your network is not currently connected to the Internet. If a connection to the Internet already exists, you will have to replace the current Internet router and apply its IP address to the BorderGuard's external Ethernet. The basic configuration for this installation option is diagrammed in Figure II-1.

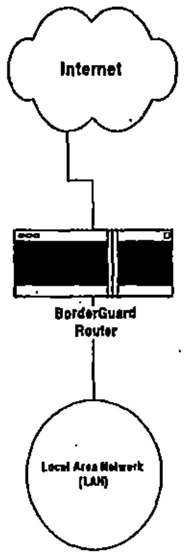


Figure II-1: The BorderGuard as the Internet Router

Options 2–5—Unable to Replace the Existing Internet Router

The remaining configuration options focus on situations where it is not possible or feasible to replace an existing Internet router with a BorderGuard unit (for example, your company's ISP may require a particular brand of router, such as a *Cisco*). The following situations are discussed:

- a class B address
- one or more unused class C addresses
- no unused class C addresses
- a class C address that cannot be subnetted

Option 2—A Class B Address

In this case, the Internet connection is based on a class B Internet address (the first number in the IP address is between 128 and 191, such as 130.130.x.x). You can assign an unused logical class C address to the interface between the current router and the BorderGuard router (such as 130.130.10.x). All current addresses can remain the same, and there is a minimal amount of overhead. This configuration is diagrammed in Figure II-2.

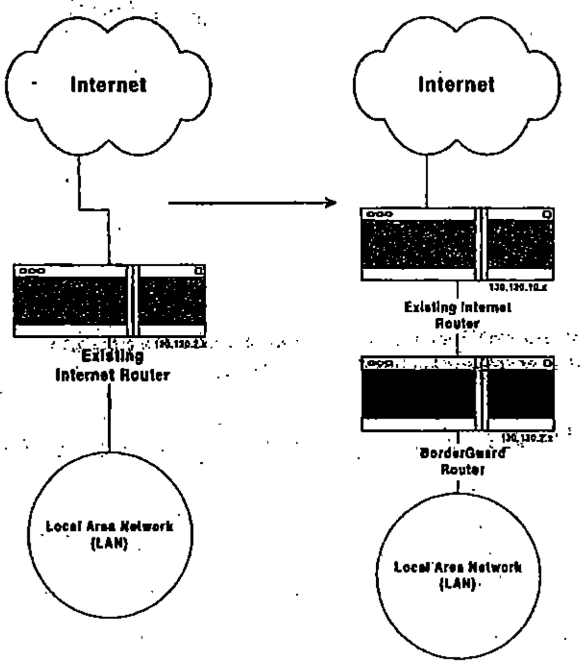


Figure II-2: BorderGuard Router Placed Behind the Existing Internet Router (Class B address space)

Option 3—An Unused Class C Address

Another option when the existing router cannot be replaced is to assign an unused class C address (the first number is greater than 191, such as 193.20.20.x) between the existing router and the BorderGuard router. This can be done if you have multiple class C addresses and at least one of them is currently unused. As with Option 2, the remaining IP addresses can remain the same. This configuration is diagrammed in Figure II-3.

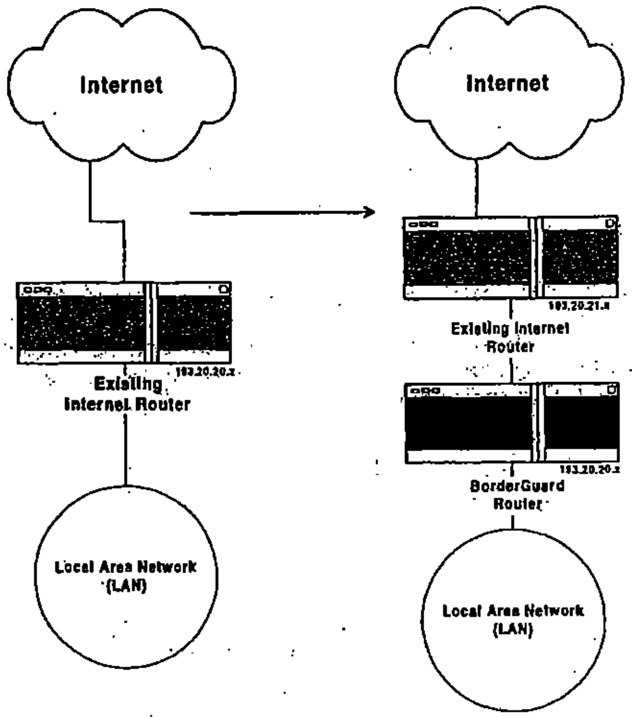


Figure II-3: BorderGuard Router Placed Behind the Existing Internet Router (Multiple Class C addresses)

Option 4—Unable to Replace Existing Internet Router and No Unused Class C Addresses

In some cases, you may be unable to replace an existing Internet router and have no unused class C addresses (the first number is greater than 191, such as 193.20.20.x). In this situation, you'll have to subnet an existing class C network and change some internal host addresses. **Always try to subnet the class C address with the least number of hosts.** The addresses that must be changed are those that end in the first subnet range, such as x.x.x.1-31 for netmask 255.255.255.224, or those that end in the last subnet range, such as x.x.x.234-254 for netmask 255.255.255.224. This configuration is diagrammed in Figure II-4.

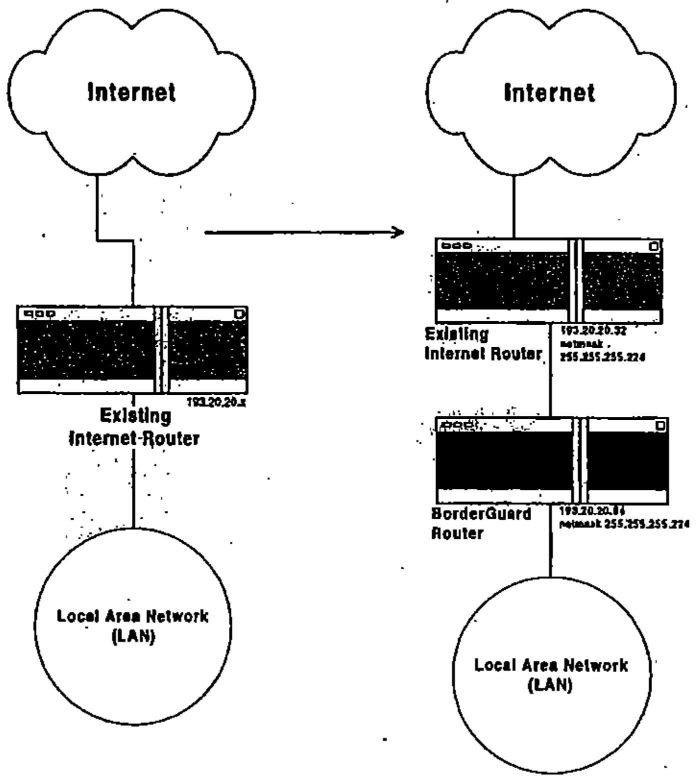


Figure II-4: BorderGuard Router Placed Behind the Existing Internet Router (Class C address is subnetted)

Option 5—Unable to Subnet The Existing Class C Address

In some instances, it is impossible to subnet a current class C address. This typically happens when you have only one registered class C address and all of its addresses are assigned. You have two options for installing the BorderGuard in this situation:

1. Obtain another class C address from the InterNic and employ Option 3 described earlier.
2. Purchase a proxy server or address translation device and remap the internal IP addresses.

The second option also requires that you subnet the class C address assigned to the BorderGuard router and the proxy server. While both of these options are fairly difficult, obtaining a new address is the least problematic of the two. These options are diagrammed in Figure II-5.

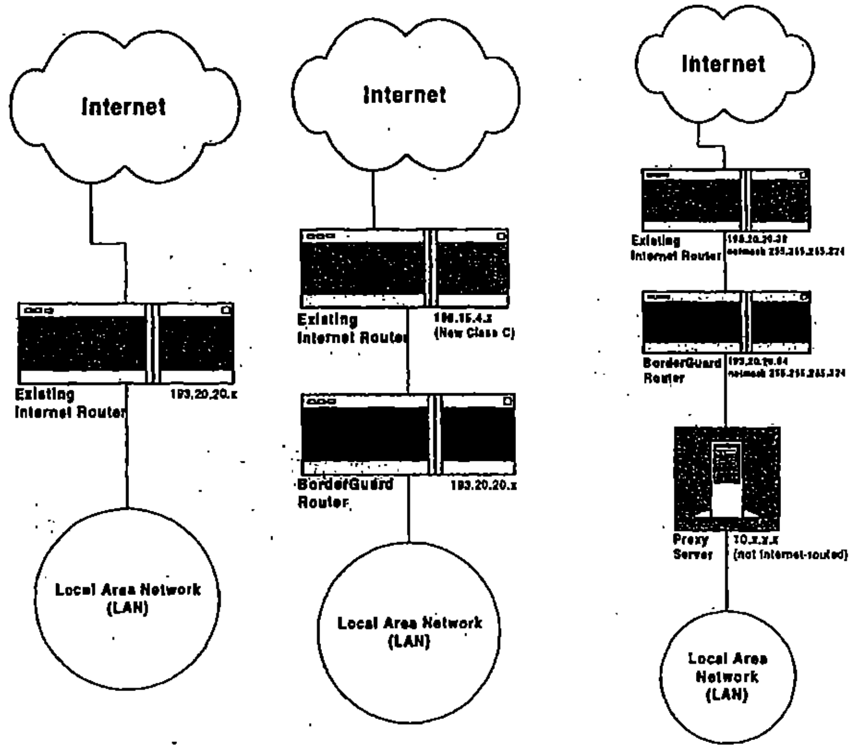


Figure II-5: BorderGuard Router Placed Behind the Existing Internet Router (Either assign a new class C address or use a proxy server for address translation)

NetRanger NSX Sensor Installation Options

You have several different options for the location of the NSX Sensor on your network(s). Each option involves different costs, such as extra ports on the BorderGuard; extra hardware (switched Ethernet hubs); and different benefits, such as increased security and performance.

Option 1—Install the NSX Sensor on a Separate, Isolated Network

This is the most secure NSX configuration. In this case, the NetRanger NSX sensor is placed on its own network. This configuration can only be implemented with the NSX 2000 or 5000. You will use one of the Ethernet interfaces on the BorderGuard 2000 to create a private network for the NSX Sensor. The benefit of this configuration is that the traffic traveling between the BorderGuard and the NSX Sensor will be isolated, which provides additional security from any inside threat. This configuration also performs slightly better than a system located on your corporate network and is similar to the diagram illustrated in Figure II-6.

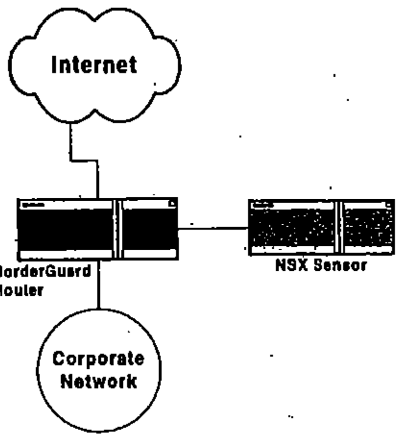


Figure II-6: The NSX Sensor Placed on a Separate, Isolated Network

Option 2—Install the NSX Sensor on the Corporate Network

In this case, the NetRanger NSX sensor resides on the internal network. This configuration can be used with either the NSX 1000, 2000 or 5000. This configuration (which is diagrammed in Figure II-7) has two disadvantages:

- Communication between the BorderGuard and NSX sensor devices is mixed in with existing internal traffic and may create some overhead on your internal network.
- The NSX sensor is vulnerable to attacks from systems within the internal network. Unlike the first option, the BorderGuard filters cannot be used to protect the NSX sensor from these types of attacks.

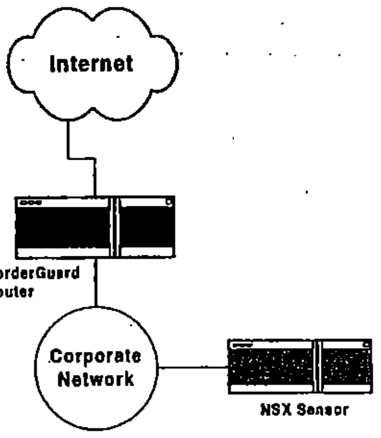


Figure II-7: The NSX Sensor Placed on your Corporate Network

Option 3—Install the NSX Sensor on a Switched Ethernet Network

With this option, the NetRanger NSX sensor resides on the same network as the rest of your users. However, the NSX Sensor is isolated from those users with a switched Ethernet hub. This configuration offers the same advantages for performance as attaching the NSX Sensor directly to the BorderGuard, and provides some additional security from the corporate network. The security won't be as robust as with Option 1, but the NSX Sensor traffic is protected from users on the internal network, which does provide some additional security. This configuration is diagrammed in Figure II-8.

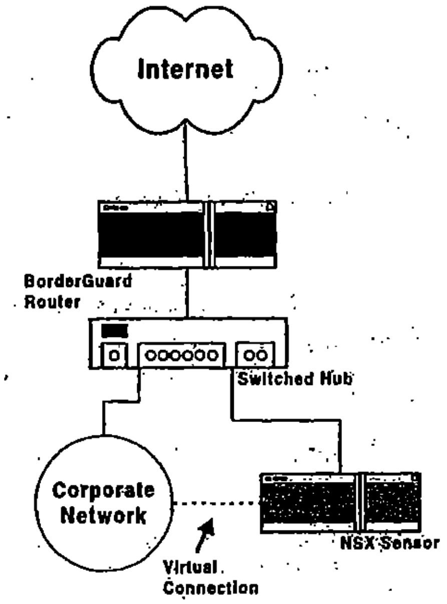


Figure II-8: The NSX Sensor Placed on a Switched Ethernet Network

NetRanger Director Setup Options

Two things must be considered when deciding where to place your NetRanger Director. The first consideration concerns operational use of the system. The Director should be placed in a physical location close to the individual(s) responsible for monitoring your networks. If you plan on using X sessions to access the Director, it should also be placed on the same network as the operators of that system. The second consideration relates to the Director's accessibility to NSX systems. There must be a path between the NSX and the Director for the alarm and management functions to work properly. If the Director is going to be placed on a network behind a proxy firewall or an address translation device, a NetRanger post office daemon must be loaded onto that device for communication to take place. Contact your WheelGroup representative for details on supported firewalls and operating systems.

Physical Installation Considerations

Because NetRanger is a significant component of your overall security environment, the NetRanger system(s) should be placed in a secure room. NSX and BorderGuard systems are shipped with rack mounts and can be located near other network equipment.

Power

Both the NSX and Director systems are currently UNIX-based and should be installed on an Uninterruptable Power Supply (UPS) to protect them from power outages and surges.

Space Considerations

The BorderGuard 1000 has a height of 1.74 inches and the BorderGuard 2000 has a height of 6.5 inches and has a standard rack-mount width and depth.

Cabling and Setup

After you have configured the NSX Sensor, you can establish a terminal connection between it and the BorderGuard router via a direct connect cable from the NSX serial port to the BorderGuard router console port. Refer to Table II-1 for the Cable and Ethernet transceiver type for your network connections. (The cable and Ethernet transceiver type will vary depending on the network connection.)

Cable Requirements

- The cable required for a BG1000-to-NSX connection is a 9-pinM to 9-pinF. (Attach a 9-pinM to the console port on the NSC and the 9-pinF to the serial port on the NSX.)
- The cable required for the BG2000-to-NSX connection is a 25-pinM to 9-pinF. (Attach a 25-pinM to the console port on the NSC and the 9-pinF to the serial port on the NSX.)

Initial Setup

To log into the BorderGuard Router after establishing the cable connection to the NSX Sensor, follow these steps:

1. **After you have attached the console cable between the NSX Sensor and the BorderGuard Router, type the following at the NSX command line prompt:**

   ```
   ➢ tip hardwire2 <return>
   ```

2. **Type the password at the router password prompt. If no password has been set, press the Enter key and you will be logged into the router.**

Interface	Connector
FDDI/single-mode FDDI	MIC / ST
IEEE 802.3/Ethernet	15-pin Ethernet AUI (10Base5)
MONITOR/MAINTENANCE ports	25-pin D-type, RS-232
Synchronous serial interface card	25-pin V.35, RS-422
Optical bypass Switch	6-pin DIN

Table II-1: Cable and Ethernet Transceiver Type for Network Connections

III Configuration and Installation

Installing and Configuring NetRanger 1.1

Proper installation of NetRanger involves the following steps:

- **Defining a Security Policy**
- **Gathering Network and Security Information**
- **BorderGuard Installation and Configuration**
- **NSX Sensor Installation and Configuration**
- **Director Installation and Configuration**

Define the Security Policy

Your security policy defines what type of activities and services you want to allow and disallow at key access points on your network. This includes:

- services to allow in from untrusted sites,
- services to allow out from internal users, and
- additional services you want to track, such as Web traffic or FTP usage.

Implementation of your security policy is based largely on the filters installed on your BorderGuard packet filter device. Use the following sections on **ICMP**, **TCP**, and **UDP** traffic as a guide to the configuration of the BorderGuard filter templates. Each section contains a subsection for **trusted** and **untrusted sites.** For more information on these services you may want to refer to such classic texts as *Internetworking With TCP/IP—Volume 1* by Douglas E. Comer.

Gather General Security/Network Information

Before installing and configuring the NetRanger system, you should gather the following information:

- BorderGuard IP addresses (One for each interface)
- NSX IP address
- Director IP address
- Internal Web server address

- Internal DNS server address
- Internal FTP server address

The following information should be available for each configured NetRanger NSX system:

ICMP (Internet Control Message Protocol)

The ICMP service allows routers and hosts to send error or control messages to other routers or hosts. One of the most frequent uses of this service is in support of the "ping" command, which queries a remote host or network device to see if it is alive on the network. This service is commonly used by hackers to discover potential targets by mapping out a remote network. It also allows internal users to check connectivity to a remote site. Use the following chart to help map allowable ICMP messages.

Trusted: Enter "A" for *Allowed* or "B" for *Blocked* (the recommended entry is in parentheses). The number beside the Type Fields indicates the offset that will be required as part of a filter to block out this particular service (see the *Editing Default Filter Templates* section for more information).

Echo Reply:	_______	(A)—Type Field (0)
Destination Unreachable	_______	(A)—Type Field (3)
Source Quench	_______	(A)—Type Field (4)
Redirect (change a route)	_______	(A)—Type Field (5)
Echo Request	_______	(A)—Type Field (8)
Time Exceeded for a Datagram	_______	(A)—Type Field (11)
Parameter Problem on a Datagram	_______	(A)—Type Field (12)
Timestamp Request	_______	(A)—Type Field (13)
Timestamp Reply	_______	(A)—Type Field (14)
Address Mask Request	_______	(A)—Type Field (17)
Address Mask Reply	_______	(A)—Type Field (18)

Untrusted: Enter "A" for *Allowed* or "B" for *Blocked* (the recommended entry is in parentheses). The number beside the Type Fields indicates the offset that will be required as part of a filter to block out this particular service and will be explained more in the *Editing Default Filter Templates* section.

NOTE

All incoming requests to your network should be blocked.

Echo Reply:	_______ (A)—Type Field (0)
Destination Unreachable	_______ (A)—Type Field (3)
Source Quench	_______ (A)—Type Field (4)
Redirect (change a route)	_______ (A)—Type Field (5)
Echo Request	_______ (B*)—Type Field (8)
Time Exceeded for a Datagram	_______ (A)—Type Field (11)
Parameter Problem on a Datagram	_______ (A)—Type Field (12)
Timestamp Request	_______ (B*)—Type Field (13)
Timestamp Reply	_______ (A)—Type Field (14)
Address Mask Request	_______ (B*)—Type Field (17)
Address Mask Reply	_______ (A)—Type Field (18)

*You may want to unblock services destined for your Internet servers such as a Web server, mail server, or FTP server. This function is explained in greater detail in the Filter Editing section.

TCP (Transmission Control Protocol)

This protocol is the heart of most of the well-known Internet services, such as WWW, e-mail, and FTP. Because a connection is established every time a TCP service is used, it is easy to ***block certain services from entering*** your network while at the same time ***allowing outgoing*** traffic. Use the following chart to help map allowable TCP services. Rather than trying to be all-inclusive, this list presents the most common TCP services that are included in the filter templates. Services that can be dynamically added to a filter have a corresponding entry in the NSX's *sensord.conf* file. The configuration of this file is discussed in a later section.

Trusted: Enter "A" for *Allowed* or "B" for *Blocked* (the recommended entry is in parentheses).

FTP Reply (Source Port 20)	_______ (B*)
FTP (Port 21)	_______ (A)
Telnet (Port 23)	_______ (A)
SMTP (Mail, Port 25)	_______ (A)
DNS (Port 53)	_______ (A)
Gopher (Port 70)	_______ (A)

Finger (Port 79)	______ (A)
WWW (Port 80)	______ (A)
POP2 (Mail, Port 109)	______ (A)
POP3 (Mail, Port 110)	______ (A)
RPC (Port 111)	______ (A)
Auth (Port 113**)	______ (B)
NNTP (News Port 119)	______ (A)
NTP (Time Port 123)	______ (A)
Exec (Port 512)	______ (B)
Login (Port 513)	______ (B)
Cmd (Port 514)	______ (B)
Printer (Port 515)	______ (A)
ntalk (Port 518)	______ (A)
uucp (Port 540)	______ (A)
X11 (Ports 6000-6063***)	______ (B)

*Source port of 20 should only be allowed going out your trusted interface if you are maintaining an FTP server on your network. This port should be restricted to your FTP server.

**Connections outgoing to port 113 are usually initiated by mail or Web servers. These services request information from remote systems concerning the user who sent an e-mail or accessed a web page. The information that is received from the remote site is untrusted and is often used by hackers to "trick" your local system into performing actions under the hacker's direction. Unfortunately, because many mail servers require this service to operate, you will need to allow this service. Upgrade or reconfigure your mail server as soon as possible, so that it no longer requires this service.

NOTE

***Often when hackers break into remote systems, they initiate X sessions back to their remote site. For this reason, it is usually not advisable to allow outgoing X connections unless absolutely necessary.

Untrusted: Enter "A" for Allowed or "B" for Blocked (the recommended entry is in parentheses). Additional space after parentheses is for the IP address(es) of the host you want to allow the service to.

FTP Reply (Source Port 20)	______	(A All IPS allowed FTP*)	______
FTP (Port 21)	______	(A 1 IP**)	______
Telnet (Port 23)	______	(B)	______
SMTP (Mail, Port 25)	______	(A 1 IP**)	______
DNS (Port 53)	______	(A 1 IP**)	______
Gopher (Port 70)	______	(B)	______
Finger (Port 79)	______	(B)	______
WWW (Port 80)	______	(A 1 IP**)	______
POP2 (Mail, Port 109)	______	(B)	______
POP3 (Mail, Port 110)	______	(B)	______
RPC (Port 111)	______	(B)	______
Auth (Port 113)	______	(A 1 or more IP***)	______
NNTP (News Port 119)	______	(B)	______
NTP (Time Port 123)	______	(B)	______
Exec (Port 512)	______	(B)	______
Login (Port 513)	______	(B)	______
Cmd (Port 514)	______	(B)	______
Printer (Port 515)	______	(B)	______
ntalk (Port 518)	______	(B)	______
uucp (Port 540)	______	(B)	______
X11 (Ports 6000-6063)	______	(B)	______

*TCP Source Port 20 must be allowed to all hosts that need to be able to "ftp" to the Internet. Most of the time, this includes everyone, so this TCP service is allowed through.

**For these services, it is usually best to restrict them to the one host providing this service, i.e., Web server, mail server.

***The authentication service needs to be allowed for connection back to the mail server and sometimes to individual hosts. Many mail services require this service to pass before they will accept mail, so any trusted system sending mail must allow this service back to it. Additionally, some Web sites require this same service. Allowing everyone to use this service does not create any known security holes.

UDP (User Datagram Protocol)

Very few UDP services should be allowed between your network and untrusted sites. UDP is a connectionless service, meaning that it is impossible to distinguish between session initiation and general session data. Use the following chart to help map allowable UDP services. This list is not all-inclusive, but rather lists the most common UDP services that are included in the filter templates. Other services can be added to the filter and will need to have a corresponding entry in the *sensord.conf* file. The configuration of this file is discussed in a later section.

Trusted: Enter "A" for *Allowed* or "B" for *Blocked* (the recommended entry is in parentheses).

DNS (Port 53)	_______ (A)
TFTP (Port 69)	_______ (B)
RPC (Port 111)	_______ (B)
NTP (Port 123)	_______ (A)
Netbios (Ports 137-139)	_______ (B)
SNMP (Ports 161,162)	_______ (B)
Syslog (Port 514)	_______ (B)
RIP (Port 520)	_______ (B*)
Response ports (Ports > 1023)	_______ (A**)
Response ports (Ports > 1023)	_______ (A**)

*If you are using NetRanger between corporate partners or internal business networks, you may need to allow RIP to pass. Work with your network administrator to determine what protocols need to be allowed.

**Some services such as DNS require high-destination UDP ports to reply to requests. We recommend that when you allow such ports, you restrict it to replies to certain services. For example, DNS responses should allow a destination port > 1023 if and only if the source port is 53. For some services, such as TFTP, high-level ports must be allowed in both directions to the hosts using this service. In general, we recommend not turning on the high level-ports unless you know the specific services you are trying to allow through.

Untrusted: Enter "A" for *Allowed* or "B" for *Blocked* (the recommended entry is in parentheses). Additional space after parenthesis is for the IP address(es) of the host you want to allow the service to.

DNS (Port 53)	______	(A*) ____________________
TFTP (Port 69)	______	(B) ____________________
RPC (Port 111)	______	(B) ____________________
NTP (Port 123)	______	(B) ____________________
Netbios (Ports 137-139)	______	(B) ____________________
SNMP (Ports 161,162)	______	(B) ____________________
Syslog (Port 514)	______	(B) ____________________
RIP (Port 520)	______	(B**) ____________________
Response ports (Ports > 1023)	______	(A***) ____________________

*You will typically want to restrict external access to a single DNS server.

**If using NetRanger between corporate partners or internal business networks, allow RIP or other routing protocols to pass. Work with the network administrator to determine which protocols need to be allowed.

***The only standard high-level UDP port usually allowed from an untrusted site is back to the DNS server. Restrict it to that single IP address and to a UDP source port of 53.

BorderGuard Configuration

This section explains the settings for all of the filter (fil), command (cmd), and data privacy facility (dpf) files required to deploy a BorderGuard packet filter device. Please note that much of the information in this section is automatically set for you by the nsxinstall script shipped with NetRanger under the /usr/nr/lib directory. It is presented here as background in the event you need to manually edit these files.

Standard Files

The standard files that must be configured on a BorderGuard device are as follows:

- startup
- filters.cmd
- sleeves.dpf & sleeves.fil
- pubkeys.dpf
- shun.fil
- first.fil
- incom(x).fil
- last.fil

In order to properly understand the explanations for each of these files, use the appropriate NSC Reference Manual.

startup—This is the boot configuration file that is read each time the BorderGuard powers up or resets. This file performs the following tasks:

- naming the device,
- compiling and applying filters,
- configuring interfaces,
- setting up route tables,
- executing the files required to establish encrypted sleeves,
- establishing a nameserver,
- starts daemons such as *telnetd*, *gated*, and *snmpd*,
- and other similar functions.

filters.cmd—This file compiles all the filters that enforce your security policies. It is called in the startup file via the following command:

```
@filters.cmd
```

This file contains a single line for each filter used on the BorderGuard. Most filters.cmd files contain at least the following three entries:

```
ip compile_filter first.fil

ip compile_filter incom1.fil

ip compile_filter shun.fil
```

DPF Files

Refer to the NSC manual, *Data Privacy Facility—DPF Administrator's Guide* before attempting to make changes to any of the files in this section.

sleeves.dpf—This file contains sleeve definitions for the BorderGuard's Data Privacy Facility (DPF). During router initialization it is loaded in as a set of sleeve definition commands and placed into the DPF interface's working storage. The default sleeve settings should rarely need editing. The only entries that should be edited are sleeve definitions. The recommended format for these entries is

```
s_<netid>_<netid>_<sleeve_num>
```

where **netid** corresponds to the network ID established in the /usr/nr/etc/hosts file discussed in a later section and **sleeve_num** corresponds to the sleeve number at a particular site. The IP addresses assigned for group1 and group2 should correspond to the router interface designated on the BorderGuard as the DPF interface.

NOTE

Sleeve definitions should be identical at both the local and remote site. For example, consider site #1 with 2 NetRangers, whose routers have the DPF IP addresses 199.99.99.1 and 199.99.100.1, and site #2 with one NetRanger, whose router IP address is 198.98.98.1.

The following sleeve definitions might apply:

```
sleeve s_100_101_1 group1 199.99.99.1 group2 198.98.98.1

sleeve s_100_100_1 group1 199.99.99.1 group2 199.99.100.1
```

sleeves.fil—This file contains two filters for every DPF sleeve connection defined in sleeeves.dpf. The first filter controls outbound traffic and the second filter checks inbound traffic.

NOTE

The **IP address of the NSX** (after the *copy_to* statements) and the **sleeve name** must match the name designated in sleeves.dpf. For example, if site #1 has a class C address 199.99.99.0 and site #2 has a class C address 198.98.98.0, and the sleeve name is s_100_101_1, the filter would look something like the following:

```
filter dpf_100_101_1

ip_sa in (199.99.99.*)

set_sleeve s_100_101_1;

end;

filter dpf_fail_100_101_1

ip_da in (199.99.99.*)

not sleeve s_100_101_1

icmp_unreachable

copy_to IP_ADDRESS_NSX fail;

end;
```

A corresponding filter would be created for the remote site and would replace 199.99.99.* with 198.98.98.* but the sleeve names would remain the same.

sleeves.cmd—This file compiles and applies all the filters required to establish encrypted sleeves between a local BorderGuard and one or more remote sites. The filters are established in the *sleeves.fil* file and should be applied so that all traffic traveling between two sites is encrypted. For example, if site #1 had a class C address 199.99.99.0 and site #2 had a class C address 198.98.98.0, then the apply statements would be similar to the following:

```
ip apply dpf_100_101_1 on 199.99.99.* to 198.98.98.*

ip apply dpf_fail_100_101_1 on 199.99.99.* to 198.98.98.*
```

In this example, `dpf_100_101_1` and `dpf_fail_100_101_1` correspond to the filter names established in the file *sleeves.fil*.

pubkeys.dpf—This file contains the public keys for the router. This file should have an entry for the local router and each remote BorderGuard for which an encrypted sleeve is maintained.

NOTE

sleeves.dbf requires long keys. Be sure that you generate long public keys for both ends of the encrypted sleeve.

Filter Templates

These files establish filtering on the BorderGuard and enforce security policies. Refer to the section on the Packet Control Facility (PCF) in your NSC reference manual for information on how to create and apply filters. When you are editing these filters, you may find it helpful to refer back to the section that helped to establish a security policy.

shun.fil—NSX system uses this file to **dynamically block** outgoing as well as incoming IP addresses.

NOTE

The name of this file should NOT be changed. The only thing that needs to be edited in this file is the IP address of the NSX that follows the *copy_to* command. If you have run the NetRanger installation script *nsxinstall* this should already be changed.

first.fil—This file identifies which network traffic should be copied to the NSX sensor. First change the IP address of the NSX that follows the *copy_to* command if that has not already been changed. Then change any string-matching requirements defined in sensord.conf (described further on in this section).

For string matching to be effective, all packets for a given service (i.e. Telnet at port 23) must be copied to port 35398. By default, FTP (21), Telnet (23), mail (25), DNS (53), and RPC (111) packets are all copied. To set up string matching on additional services such as WWW (80), or on FTP data (20), add the following command lines to first.fil:

```
# Copy certain TCP traffic
ip_protocol in (6)
tcp_dp in (21,23,25,53,111,80,20)
copy_to IP_ADDRESS_NSX 35398 break;
```

NOTE

The port number following the NSX is 35398 instead of 35399 as it appears in the other filter files. **This is the UDP port for intrusion detection. Do not change it!**

incom(X).fil—This file is the heart of the NetRanger security system, and it establishes the security policy defined in the earlier section. Apply this filter to each network interface in order to establish a security policy. The general naming convention is *incom1.fil* for interface 1, *incom2.fil* for interface 2, and so on.

NOTE

The default templates assign incom1 to the interface connected to your untrusted networks. Notice that the number following the *copy_to* command in this file is 35399 instead of 35398, as indicated in the previous filter file. **This is the security policy monitoring port. Do not change it!**

If *nsxinstall* has not been run, the first piece of information that should be filled in is the IP address of the NSX after every *copy_to* command.

```
        copy_to IP_ADDRESS_NSX 35399 break;
```

Next, make changes to the *incom(X)_ip_spoof_fail* filter. **This filter prevents and alarms any connections from outside your trusted network that have a source address from your internal network.** When applied to your trusted network interface it assures that no outgoing traffic is spoofing an external address. Collect all of your internal class C and class B addresses and create an entry for each one. You should also create an entry for the individual external IP address. In the case of a site with two internal protected networks with class C addresses 199.99.99.0 and 199.99.100.0, and a router with an external IP address of 199.99.101.1, the filter attached to the untrusted network would look like the following:

```
        filter incom1_ip_spoof_fail

        ip_sa mask 0xFFFFFF00 in (199.99.99.*)

        copy_to IP_ADDRESS_NSX 35399 break

        icmp_unreachable fail;

        ip_sa mask 0xFFFFFF00 in (199.99.100.*)

        copy_to IP_ADDRESS_NSX 35399 break

        icmp_unreachable fail;

        ip_sa mask 0xFFFFFFFF in (199.99.101.1)

        copy_to IP_ADDRESS_NSX 35399 break

        icmp_unreachable fail;

        end
```

For the filter attached to the trusted network , make the following changes:

```
        filter incom1_ip_spoof_fail

        not ip_sa mask 0xFFFFFF00 in (199.99.99.*)

        not ip_sa mask 0xFFFFFF00 in (199.99.100.*)

        copy_to IP_ADDRESS_NSX 35399 break

        icmp_unreachable fail;

        end
```

The next change is based on choices made when determining a desired security policy. For each service in the security policy that is allowed, edit the current filter with its name. There are two basic cases for allowable services. The first and simplest case is when to allow all services of this type through. This is most often the case for the interface attached to a trusted network, because users want unrestricted acess to e-mail and WWW services. For example, to allow all outgoing mail traffic, simply comment out the body of the filter *incom(X)_smtp_fail* as follows:

```
filter incom2_smtp_fail
#tcp_dp in (25)
#copy_to IP_ADDRESS_NSX 35399
#icmp_unreachable fail;
end
```

This allows all outgoing connections to port 25.

The second case for allowing a given service involves restricting that service to a specific destination or source IP address. This type of restriction is usually applied to the interface attached to an untrusted network. In this case, the filter needs to be edited as follows:

```
filter incom1_icmp_fail
ip_protocol in (1)
```

This is where you should enter the typefields of the ICMP services you # want to block. In this example, we are blocking "echo requests" type field 8.

To block "timestamp requests" you would add 13. "tl_byte 0 in (8,13)"

```
tl_byte 0 in (8)
not ip_sa in (198.98.98.*)
not ip_da in (IP_ADDRESS1, IP_ADDRESS2, IP_ADDRESS3, ETC.)
copy_to IP_ADDRESS_NSX 35399
icmp_unreachable fail;
end
```

The above example allows all ICMP echo requests that are from the network 198.98.98.0 or that are destined for IP_ADDRESS1, 2, or 3. All of the filters should comply with your security policy requirements. Please refer to the NSC reference manual corresponding to your unit for more information on this topic.

last.fil— Although this filter is not required, it provides a convenient way to permanently block a particular site. Such a filter might look as follows:

```
filter last_block_fail
ip_sa in (BLOCKED_IP_NETWORK)
copy_to IP_ADDRESS_NSX 35399
icmp_unrechable fail;
end
```

NSX Configuration and Installation

Installation of the NSX Sensor system involves configuration of the following types of files:

- Solaris OS files
- NSX system files
- Daemon configuration files

This section explains how to configure the files associated with each of these categories.

Solaris OS Files

A number of basic Solaris /etc files must be edited before a NSX Sensor *system* can be configured. These files include:

- /etc/hosts
- /etc/hostname.elx0
- /etc/defaultrouter
- /etc/netmasks
- /etc/nodename
- /etc/hosts.allow
- /etc/hosts.deny

NOTE

The default IP address of all factory-shipped NSXs is 10.1.4.204.

NSX System Files

As noted in *Appendix C*, the */usr/nr/etc* directory contains all of the NSX's system files. These files are modeled after Unix */etc* files and Unix system administrators should have no problem understanding their format and entity relationships. The files currently are as follows:

- auths
- daemons
- destinations
- hosts
- organizations
- routes
- services
- signatures

NOTE

These files are *described* in order of use rather than alphabetically.

organizations

This file identifies all of the *organizations* and their *organization ids* currently registered for a given NetRanger system. The format of the file is:

```
<org_id> <organization_name>
```

A typical file might look as follows:

```
100     wheelgroup
101     netsolve
102     btg
1001    dataworks
```

NOTE

This file serves as a *global* registry that must agree across all of the NSX and Director systems within a NetRanger domain.

hosts

This file is much like a unix /etc/hosts file. It enumerates the organizations and hosts that are recognized by a given NSX or Director system in a NetRanger configuration. Each entry has the form:

```
<host_id>.<organization_id>    <host_name>
```

where `host_id` and `organization_id` are numeric identifiers assigned by WGC, and `host_name` is a DNS-like name assigned by the user. In addition to a list of recognized hosts, this file always contains a *localhost* entry.

A typical hosts file might look like the following:

```
10.100    localhost
10.100    admin.wheelgroup
11.100    dev.wheelgroup
12.100    data.wheelgroup
13.100    nxs-1.wheelgroup
14.100    nxs-2.wheelgroup
15.100    nxs-3.wheelgroup
```

NOTE

Entries for each host must be consistent across *all* NSX and Director systems. Otherwise, some systems will generate "unrecognizable host" messages in ~/var/errors .postofficed.

services

This file defines the *application_id* for each of the daemon services found in **~/bin**. When combined with the *host_id* and *organization_id*, these identifiers uniquely identify a daemon within a NetRanger security map. Each entry has this form:

```
<application id>  <daemon name>  [<comment>]
```

All NSX hosts should have the same default services file, which is currently defined as follows:

```
10000 postofficed # Provides the NetRanger comm system
10001 sensord     # Monitors network traffic for security
10002 configd     # Sets and Gets configuration information
10003 managed     # Manages NetRanger components
10004 eventd      # Sends messages and alarms to pagers
10005 loggerd     # Logs events, cmds, errors, IP logs, etc.
10006 smid        # Routes msgs & events to nrdirmap
10007 sapd        # Stages log data into a DBMS
```

auths

This file identifies which type of *configd* operations are authorized for the each of the hosts listed in this file. Each entry follows this form:

```
<host name> <configd services>
```

In the following example, `admin.wheelgroup` has full authorization, whereas `data.wheelgroup` can only read configuration information.

```
admin.wheelgroup     GET,GETBULK,SET,UNSET,EXEC
nsx-1.wheelgroup     GET,GETBULK
nsx-2.wheelgroup     GET,SET,UNSET,EXEC
```

NOTE

Each entry must have a corresponding entry in ~/etc/hosts.

destinations

NetRanger is a distributed application that has the ability to route messages from a given post office to any of the daemon services registered in the ~/etc/hosts and ~/etc/services files. The destinations file identifies what type of messages get routed to a given application on a given host. Each entry is of the following form:

```
<Destination Id.>  <host name>  <daemon name>  <message level>  <message
type>
```

Where `<Destination Id>` is a Numeric Id. Up to 32 destinations are currently allowed. `<host name>` must be an entry from ~/etc/hosts. `<daemon name>` must be an entry from ~/etc/services. `<message level>` Identifies the level of messages that will be routed. Messages below that level are not passed on to *postofficed*. `<message type>` identifies the types of messages to route.

For example, in the following destinations file, level 2 messages of the type *event, error, command,* and *IP_log* are being routed to the *smid* daemon on the host *admin.wheelgroup*. This type of configuration would normally be found on an NSX system, where *admin.wheelgroup* is a Director system.

```
1    admin.wheelgroup    smid    2 EVENTS,ERRORS,COMMANDS,IPLOGS
```

routes

This file identifies the actual IP routes that *postofficed* uses to send messages between different hosts. Each entry is of the following form:

```
    <host name>  <connection #>  <IP address>  <UDP port>  <Type>
```

`<host name>` must be an entry from ~/etc/hosts. `<connection #>` identifies the priority of this entry relative to the other routes enumerated in this file. The lower the number, the higher the route priority. If there is more than one entry of the same priority, *postofficed* accepts the last entry of that priority. `<IP address>` identifies the **end-point** IP address of the **remote system.** `<UDP port>` identifies the UDP port service to route through on each host. These ports are usually in the 30-50000 range. `<Type>` identifies whether the connection is a dial-up vs. dedicated connection. This field is currently not used.

In the following example, `nsx-2.wheelgroup` serves as the primary route to the IP Address 10.1.7.12, and `nsx-3.wheelgroup` is the secondary route to the same host.

```
        nxs-1.wheelgroup     1   10.1.7.9 45000     1

        nxs-2.wheelgroup     1   10.1.7.12    45000     1

        nxs-3.wheelgroup     2   10.1.7.12    45000     1
```

daemons

This file contains the names of daemons that should be started when the system starts up. Each entry is of the form:

```
        <daemon name>
```

For example, the following sample entry would start *postofficed, configd, loggerd,* and *smid* on a Director system:

```
        nr.postofficed

        nr.configd

        nr.loggerd

        nr.smid
```

NOTE

Whereas the *services* file identifies all of the daemon services a given NSX or Director system is *capable* of running, the *daemons* file identifies the services to launch upon startup.

signatures

This file associates *signature ids* with *signature names* for NetRanger applications, and is used primarily by Director systems. The format of each entry is in the file is:

```
<signature id> "<signature name>"
```

For example, the following sample entry defines a few attack signatures.

```
1000 "Bad Option List"
1001 "Record Packet Rte"
1002 "Timestamp"
1003 "Provide s,c,h,tcc"
1004 "Loose Src Rte"
1005 "SATNET ID"
```

NOTE

This file should NOT be modified.

Daemon Configuration Files

There is a one-to-one correspondence between daemons and their configuration files. Whereas the naming convention for a daemon is nr.<daemon>, the convention for its configuration file is <daemon>.conf. Each of these files contains token-based configuration information of the form: <token> <value>. For example, the error file for *nr.managed* is identified in *managed.conf* as follows:

```
FilenameOfError    ../var/errors.managed
```

Although these entries can be set via a text editor, all changes should be managed via **nrConfigure.** Otherwise, the chance of generating errors increases.

- *configd.conf*
- *eventd.conf*
- *loggerd.conf*
- *managed.conf*
- *postofficed.conf*
- *sapd.conf*
- *sensord.conf*
- *smid.conf*

With the exception of *managed.conf*, *sensord.conf*, and *smid.conf* these files currently contain little more than timing intervals and the name of the ~/var/error file for each daemon service. The remainder of this section is devoted to explanations of *managed-*, *sensord-* and *smid.conf.*

managed.conf

This file is used by the NetRanger NSX to configure the management daemon that controls actions on the router. Two types of tokens in this file require special attention: *NetDevice* and *NeverShunAddress*.

NetDevice identifies the IP address and the system password of a NSC BorderGuard the NSX Sensor is connected to. A separate *NetDevice* entry is required for every BorderGuard system being managed by the NSX system. The format of this token is as follows:

```
NetDevice    ROUTER_IP_ADDRESS       NSC_Borderguard_v1  ROUTER_PASSWORD
```

NeverShunAddress identifies the IP address of a host or network device you never want the NSX to BorderGuard to shun. At the very minimum you make sure that this file contains *NeverShunAddress* entries for the local system and the remote NSX or Director system. Other addresses can be added based on your operational needs. Example entries might look as follows, where <255.255.255.255> identifies the subnet mask for the IP address you do not want shunned.

```
    NeverShunAddress ROUTER_IP_ADDRESS       255.255.255.255

    NeverShunAddress NSX_IP_ADDRESS          255.255.255.255

    NeverShunAddress INTERNAL_NETWORK        255.255.255.255
```

NOTE

NeverShunAddress entries do not bypass existing filters, but rather prevent them from being completely blocked out of the router.

sensord.conf

This file serves as the heart of the NSX Sensor system, and is the largest of all of the configuration files. It can be logically broken down into the following sections:

- General
- Event Specification
- Strings To Look For
- Matched String and Event Levels
- TCP/UDP Ports and Event Levels
- Policy Violations
- Excluded Events

General

At a minimum, you will need to edit the following lines:

```
RecordOfInternalAddress   INTERNAL_NETWORK_ADDRESS INTERNAL_NETWORK_NETMASK
RecordOfInternalAddress   ROUTER_EXTERNAL_IP_ADDR   255.255.255.255.
```

These two lines establish which networks NetRanger will protect. **It is important to include all internal addresses because this information is also used for intrusion detection.** You should have one entry for each network address and one for the router external interfaces.

```
    RecordOfDataSource      ROUTER_IP_ADDRESS    255.255.255.255
```

This line determines which BorderGuard the sensor daemon will receive information from. You need to fill in the router IP address for this line.

```
    RecordOfLogAddress    LOGGED_NETWORK_ADDRESS    255.255.255.0
```

This line establishes a binary log of a particular host or network. This is most commonly used to support an investigation and creates a binary record of all data originating from a particular source address. Initially, this line will probably be commented out.

```
    MinutesOfAutoLog       LOG_MINUTES
    MinutesOfAutoShun      SHUN_MINUTES
```

These two lines establish the amount of time that NetRanger will shun or log after a particular alarm is received.

Event Specification

The first task-specific segment is "Event Specification", which is based upon the SigOfGeneral token. This token identifies an *action* and *alarm levels* for various types of events. *sensord.conf* contains a number of different `SigOf-` tokens, all of which share the same basic format, which is defined as follows:

```
SigOfGeneral  SIGID   ACT   D1   D2   D3   D4   # Description of event
```

NOTE

`SIGID` is a numeric identifier that establishes a logical link to internal signature ids in the *sensord* daemon, and should be treated as **read-only** information. Any changes or additions to these entries will disrupt NSX functionality.

The first column that can be configured is the `ACT` column. It contains a number that specifies how to respond (an *action*) to a particular event. Valid `ACT` settings include:

0 - no action

1 - shun

2 - log

3 - shun and log

The remainder of SigOfGeneral maps alarm levels to each of the destinations specified in /usr/nr/etc/destinations. This makes it possible to uniquely define the severity of an event for every destination the NSX is configured to talk to.

The following SigOfGeneral defines what to do when a `TCP Port Sweep` is encountered. It first states that no automatic action should be taken (`ACT=0`). It also defines that level "5" alarms should be sent to its first three destinations, and a level "4" alarm to the last destination.

```
        SigOfGeneral   3001    0    5   5    5    5   # TCP port sweep
```

Strings To Look For

This section defines the strings to look ror within a given TCP/IP service. The general format for an entry in this section is the following:

```
RecordOfStringName  StringID   TCP/IP Port   Direction   Num Occurances  "Matched String"
```

`StringID` establishes the unique ID for that particular string. `TCP/IP port` tells NetRanger which port to watch to look for this string. Note that *first.fil* must have a *Copy_To* entry for this port that copies all of its packets to the NetRanger NSX. `Direction` tells the NSX which direction to look for this string;:

1 - external-to-internal

2 - internal-to external

3 - both directions

The number of occurrences indicates how many times the NSX must see this string before it alarms. `Matched String` is the actual string to be matched. The exact format is controlled by regular expression (regex) syntax.

Matched String and Event Levels

This section identifies action and alarm levels for the string matches defined in the previous section. `SigOfStringMatch` uses the same format as `SigOfGeneral` described above, and is linked to a `RecordOfStringName` by its `StringID`.

```
SigOfStringMatch  StringID   ACT   D1   D2   D3   D4   # Description of event
```

TCP/UDP Ports and Event Levels

This section establishes the alarm levels for any packet, successful or not, with a TCP or UDP destination port corresponding to the entry. The action fields and the event level specification are the same as in previous sections.

The following entry illustrates that all Telnet attempts (Port 23) should have no action taken but should generate a level 2 alarm:

```
               <port> <action> <dest1> <dest2> <dest3> <dest4>

SigOfTcpPacket    23       0       2       2       2       2
```

The next entry indicates level 3 alarms for all TFTP (port 69) attempts which logs the packets:

```
               <port> <action> <dest1> <dest2> <dest3> <dest4>

SigOfUdpPacket    69       1       3       3       3       3
```

Policy Violations

This section identifies the filter names and IDs for all BorderGuard filters that can pass information to the NSX. In the following example, `Income_1_Filter_Fail` has a filter ID of `1000`.

```
        RecordOfFilterName  1000   Income_1_Filter_Fail
```

Each of these policy filters has a matching SigOfFilterName that identifies its action and destinations. The format is identical to the SigOf- tokens described above. In the following example a `Income_1_Filter_Fail` event will be logged to the second destination defined in /usr/nr/etc/destinations.

```
           <filter_id> <action> <dest1> <dest2> <dest3> <dest4> —>

SigOfFilterName   1000      2       2       2       2       2
```

Excluded Events

The final section of */usr/nr/etc/sensord.conf* is the "Excluded Events" segment. This is where alarms can be *locked out* for events at specific source addresses. The standard format follows:

```
        RecordOfExcludedAddress   SigID   SubSigID    IP Address
```

`SigID` corresponds to the primary signature established in the "Event Specification" section that establishes all of the primary IDs. `SubSigID` corresponds to those signatures established in the latter sections, such as those for filtering and string matching.

smid.conf

This file is the configuration file for *sapd*, which passes information to the Director *nrdirmap* application for display within the OpenView user interface. The only token that needs to be edited in this file is `DupDestination`, which defines duplicate destinations for the events. This token allows a Director system to forward event information onto other services and systems. In the following example, events, commands, and errors of level "1" and above are forwarded onto *loggerd*. This allows the Director to monitor ***and*** log events based on a single NSX data stream. This in turn, helps reduce the amount of network traffic a NSX must transmit to a Director. Instead of having to transmit two identical (but separate) data streams to `director1.wheelgroup`, the NSX is able to send a single stream that is duplicated on the Director platform.

```
DupDestination  director1.wheelgroup  loggerd  1  ERRORS,COMMANDS,EVENTS
```

As the next example shows, this token can also be used to forward events onto another Director system. In this case, however, only events of level "2" and above are forwarded.

```
DupDestination  director2.wheelgroup  smid  2  ERRORS,COMMANDS,EVENTS,IPLOGS
```

NOTE

The host and organization IDs must match entries defined in /usr/nr/etc/hosts.

Event Signatures

Event signatures consist of a unique signature identifier (SigID) and a sub-signature identifier (SubID). This section includes a list of the event signatures currently defined in NetRanger. SigIDs are built into NetRanger; SubIDs change depending upon the SigID they come under.

TCP/IP Event Signatures

TCP/IP event signatures are currently divided into three groups: **context-, content-, regular expression-** and **NetSentry-based.**

Context-Based Signatures

Context-based signatures are based on information passed in the TCP/IP header. This can include such things as the destination port, i.e. TCP port 80 for WWW traffic, IP Options, i.e. source routing, or a combination of events found in a hacking attack. The following group of signatures are context-based attacks that are detectable using NetRanger. signatures are generated from stateful multi-header packet analysis. NetRanger detects stateful attacks by remembering a sequence of events, or by reconstructing packets to find certain strings or attacks. Those attacks for which alarming is dependent upon maintaining the state of a connection are indicated by a "*" following the attack name. These signatures currently analyze the following types of events:

- **Source Routing**—NetRanger detects both loose and strict source routing which is commonly used by hackers to bypass rules found in filtering routers.
- **ICMP Network Sweeps**—This attack uses the ICMP protocol to discover which machines are alive on a remote network. This is most often used as the first step of an attack to find potential targets. This can be implemented three different ways using different ICMP types. All three are detected within NetRanger
- **Fragmented ICMP traffic**—To get around filtering on large ICMP traffic, it is possible to fragment this traffic and "trick" some intrusion detection systems. NetRanger checks for and alarms on this activity.
- **Large ICMP traffic** —Numerous computers are vulnerable to an attack where if you send an ICMP packet with an extremely large data size it will crash the machine. NetRanger blocks and alarms this traffic.
- **TCP Port Sweep**—When targeting a specific machine, a hacker will frequently run a TCP port sweep to get a list of all available services on the remote target.
- **Half Open SYN Attack** —This attack was recently publicized when it was used to shut down several Internet Service Providers. This attack can crash a machine by overloading it with TCP connection requests that it never closes.
- **TCP Hijacking** —This attack looks for a characteristic of attacks which take over an existing TCP connection.

- **UDP Port Scan**—When targeting a specific machine, a hacker will frequently run a UDP port sweep to get a list of all available services on the remote target.
- **SATAN Scan**—This looks for both the normal and heavy SATAN attacks.

Content Based Signatures

For these attacks, NetRanger looks further than simple TCP/IP header information. It actually looks inside the packet for data which indicates an attack in progress. Most of these signatures take advantage of looking for these signatures within a certain context. For example, Sendmail attacks look for certain strings only within the sendmail port 25.

- **Smail attack** —NetRanger looks for an attack which was only found in the e-mail package "smail".
- **Sendmail invalid recipient** —This looks for attacks which try to send an e-mail to a program on a remote machine. The attack expects the remote machine to execute the e-mail as if it were a program.
- **Sendmail invalid sender** —This attack is like the previous attack except that the sender of the e-mail appears to be a program. When an error occurs within the e-mail the remote machine attempts to return it to the original sendor. The e-mail is then executed as a program on the remote machine.
- **Sendmail reconnaissance** —This attack is used to gather information about remote users through the mail port. This is usually used as a preface to other attacks.
- **TFTP password** —This looks for anyone attempting to get the password file using the TFTP service which requires no authentication.
- **DNS HINFO Requests** —This attack uses DNS to gather information about a specific host.
- **DNS Zone Transfer Request** —This attack attempts to gather information about all hosts registered with your DNS server. This can be used by hackers to try to get a map of your network.
- **DNS request for all records** —This attack requests all records maintained on a remote server and it is used to gather information for a future attack.
- **RPC port registration** —This is used to register a specific application to a port on a remote machine and should never occur across the network.
- **RPC port unregistration** —This is used to unregister a specific application to a port on a remote machine and should never occur across the network.
- **RPC dump** —This is used to query a remote machine about which services are running at what ports. This is commonly used by hackers as a preface to an attack.
- **Proxied RPC request** —This is an attack where you trick the remote machine into issuing a request to a service such as NFS for you. This makes the source address appear to be the local machine instead of the attacking machine.

- **NFS mountd request** — This is an attack where a remote user tries to connect to the mountd process. This can be used to determine what file systems a site is sharing on the network and how it might be exploited.
- **Rexd request** —This is a request to the remote execution daemon and is used to run programs on a remote machine without authentication.
- **YP attacks** —There are five separate attacks which NetRanger looks for which try to take advantage of the service which maintains network password and other files.
- **Loadmodule attack**—NetRanger looks for a string characteristic of an attack which tricks a set uid program into giving system administrator privileges to the attacker.
- **Any matched string**—NetRanger can look for any string within any service using regular expression matching. This allows any organization to define their own requirements and look for abuse of their proprietary systems.

IP Options Events

IP options consist of a variable-length list of optional information for an IP datagram. Options are rarely used and not all routers and host support them. A good security measure is to refuse routing IP datagrams with options. Detecting IP options from externally connected networks provides a good indicator of potential network problems and attacks.

1000 IP options—Bad option list

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

The list of IP options for the datagram is incomplete or malformed. This may indicate an attack where the attacker is using improperly developed hacking software.

1001 IP options—Record packet route

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This option instructs each router to record in the options list the IP address of the interface that the packet will be transmitted from. Because of the limited size of an IP header, only nine addresses can be store within this list.

1002 IP options—Timestamp

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This option is similar to the record route option. Routers are requested to add timestamps into the options list. This option is of little value because of the limited size of the options field.

1003 IP options—Provide s,c,h,tcc

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

Basic security options as defined in RFC 1038. Used to carry security level and accrediting authority flags. Implementation of security via these options is obsolete and should not be used.

1004 IP options—Loose source route

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This option specifies a list of IP addresses that the IP datagram must traverse (excluding the routers the datagram can also pass through). An attacker may transmit datagrams into a network using spoofed source addresses that appear to come from the target network. Responses to these packets are transmitted back to the attacker because the host recognizes the source route option. This allows attackers to defeat IP address based authentication mechanisms. Source route attacks usually use *loose routing* due to the number of hops a datagram must traverse across the Internet.

1005 IP options—SATNET id

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

Stream identifier option. This option is obsolete and should not be encountered.

1006 IP options—Strict source route

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This option specifies the exact list of IP addresses that the IP datagrm must traverse. This is similar to loose source routing. Source route attacks usually do not use this type of source routing.

ICMP Events

ICMP datagrams are generated to provide administrative and diagnostic information. The most common use of ICMP datagrams is the "ping" program, which verifies the existence of a host. ICMP traffic can provide much information about a network, and it can also modify network characteristics such as routing tables. Attackers use ICMP to discover and modify this information. SubIDs for the following signatures are set to zero.

2000 ICMP Echo Reply

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is generated in response to an echo request. This type of datagram is sent from the host being 'pinged'.

2001 ICMP Unreachable

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is transmitted to a host stating that the intended destination is unreachable for the IP datagram it transmitted. The first 64 bits of the failed datagram is transmitted along with the unreachable message. Unreachable messages can be used to disrupt existing TCP sessions.

2002 ICMP Source Quench

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message is transmitted to a host to report network congestion and requests a reduction in the current rate of datagram transmission. While not all systems support this feature, it can be used to enact a denial-of-service attack. The performance of the targeted host can be compromised by sending a continuous stream of these source quench messages.

2003 ICMP Redirect (change a route)

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is transmitted from a router to host with an update for the host's routing table. This enables hosts to start with the minimal routing configuration thatis subsequently updated by the network's router tables. Attacker's use redirect messages to place incorrect routes into a target host's routing table to support IP hijacking and other attacks.

2004 ICMP Echo Request

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message requests that the destination host transmit back an echo reply message. This type of datagram is sent to the host being 'pinged'. Individual requests are not a security threat. Large numbers of these requests may be a denial of service attack or network reconnaissance as depicted in SigID 2100 below.

2005 ICMP Time Exceeded for a Datagram

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message was designed to report circular or excessively long routes. A router decrements the time-to-live (TTL) counter whenever it processes a datagram and discards the datagram when the count reaches zero. The first 64 bits of the discarded datagram are transmitted along with the time exceeded message to the originating host. These messages typically occur when an attacker is using the "traceroute" program to help map out a target network. It also occurs when a host has the default TTL set to low (i.e. 30) and transmits datagrams to a destination far across the Internet.

2006 ICMP Parameter Problem on Datagram

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This message is transmitted when a datagram header is incorrect. The first 64 bits of the incorrect header is also sent.

2007 ICMP Timestamp Request

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message requests the destination host to transmit back a timestamp reply containing the host's current time. Individual requests are not a security threat. Large numbers of these requests may be a denial of service attack or network reconnaissance as depicted in SigID 2101 below.

2008 ICMP Timestamp Reply

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

The message generated in response to a timestamp request. Use of this feature could be considered as another type of "ping".

2009 ICMP Information Request (obsolete)

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This type of ICMP packet is obsolete and should not be used.

2010 ICMP Information Reply (obsolete)

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

This type of ICMP packet is obsolete and should not be used.

2011 ICMP Address Mask Request

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 2

This message asks the destination host to transmit back the address mask in use on the network. This service was designed to allow diskless clients to set the address mask by broadcasting this request over the local network. Individual requests are not a security threat. Large numbers of these requests may be a denial of service attack or network reconnaissance as depicted in SigID 2102 below.

2012 ICMP Address Mask Reply

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 1

The message generated in response to an address mask request. RFC 950 added this feature, but RFC 1122 forbids a host from sending replies unless it has been explicitly configured as an authoritative agent for address masks. Hosts that do not implement this feature correctly may respond to a targeted request - another type of "ping".

2100 ICMP network sweep w/Echo

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies a host that is transmitting ICMP echo request datagrams to multiple hosts on the network. This method is commonly used by attackers to identify active hosts within a network address range. While this can be a serious attack, network management tools such as HP OpenView also perform this type of network discovery on a regular basis.

2101 ICMP network sweep w/Timestamp

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies an attacking host that has transmitted ICMP timestamp request datagrams to multiple hosts on the network. While this request can also be used to identify active hosts within a network address range, most attackers use the more common echo request method (see SigID 2100).

2102 ICMP network sweep w/Address Mask

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies an attacking host that has transmitted ICMP address mask request datagrams to multiple hosts on the network. While this request can also be used to identify active hosts within a network address range, most attackers use the more common echo request method (see SigID 2100).

2150 Fragmented ICMP Traffic

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 4

This signature identifies an attacking host that has transmitted a fragmented ICMP packet. By design, ICMP packets are small and should never be fragmented. There are attacks that utilize fragmented ICMP packets to crash target systems.

2151 Large ICMP Traffic

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 4

This signature identifies an attacking host that has transmitted a large ICMP packet. By design, ICMP packets are small and should never be fragmented. There are attacks that utilize large ICMP packets to crash target systems.

3000 TCP Connection Logging

SubID Values: 0 - 65536 [SYN] Destination TCP port

100000 - 165536 [SYN-ACK] Destination TCP port + 100000

200000 - 265536 [FIN] Destination TCP port + 200000

300000 - 365536 [RST] Destination TCP port + 300000

Misc Field Info: TCP sequence number

Recommended Alarm Value: 1

Packets Required: TCP packets with the SYN, FIN, or RST flags set

This is event is used for logging TCP traffic. The token ***LevelOfTrafficLogging*** is used to configure the level of logging.

Level 1: No TCP logging occurs.

Level 2: Only TCP SYN packets are logged (default). This indicates that the source host has initiated an attempt to establish a TCP connection to the destination host using the TCP ports specified. The SubID for this signature is the destination TCP port. If the destination host refuses this connection because the requested port does not exist, an ICMP unreachable message will immediately follow.

Level 3: All TCP SYN, FIN, and RST packets are logged.

The sequence numbers stored in the miscellaneous field provide enough information to determine the number of bytes transferred within a TCP connection. This is accomplished by subtracting the initial sequence number from the SYN packet from the final sequence number from the corresponding FIN packet.

3001 TCP port sweep

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature identifies an attacking host that has initiated a series of TCP connections to a number of different destination ports on the target host. This method is used by attackers to determine the services available on the target host for potential exploitation.

3050 Half-open SYN attack

SubID Values: 0

Misc Field Info: Affected TCP port on target system

Recommended Alarm Value: 5

This signature identifies the targeted host and TCP port where the source address and port may be randomly generated by an attacker. Detection of this signature is currently limited to FTP, Telnet, WWW, and E-mail servers.

3100 Small attack

SubID Values: 0

Misc Field Info: "to: bounce"

Recommended Alarm Value: 4

This signature detects the very common "smail" attack against e-mail servers.

3101 Sendmail Invalid Recipient

SubID Values: 0

Misc Field Info: "to: I"

Recommended Alarm Value: 4

This signature detects any mail message that is transmitted to an address of "pipe" something. Due to the vulnerabilities previously discovered in sendmail and the complexity of the software, destination addresses of this type should not be allowed.

3102 Sendmail Invalid Sender

SubID Values: 0

Misc Field Info: "from: |"

Recommended Alarm Value: 4

This signature detects any mail message that is transmitted with a return address of "pipe" something. This should not be allowed under any circumstance.

3103 Sendmail Reconnaissance

SubID Values: 0

Misc Field Info: "vrfy" or "expn"

Recommended Alarm Value: 2

This represents a reconnaissance attempt by an intruder. It may also represent an advanced user attempting to determine the mailing address of a friend or co-worker.

3104 Archaic Sendmail Attacks

SubID Values: 0

Misc Field Info: "wiz" or "debug"

Recommended Alarm Value: 2

This sendmail attack is archaic and should not work against current versions of sendmail.

3200 WWW phf attack

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This serious attack is used to exploit the phf program released with the NSCA and Apache web servers.

3201 WWW General cgi-bin attack

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature detects any cgi-bin script that attempts to retrieve the file /etc/passwd.

3250 TCP Hijacking

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature analyzes both streams of data within a TCP connection to detect when TCP hijacking may have occured. This current implementation of this signature does not detect all types of TCP hijacking and false positives may occur. Even when hijacking is discovered, little information is available to the operator other than the source and destination addresses and ports of the systems being affected.

4000 UDP packet

SubID Values: 0 - 65536 Destination UDP port

Misc Field Info: none

Recommended Alarm Value: 1

This is event is used for logging UDP traffic. The token ***LevelOfTrafficLogging*** is used to configure the level of logging.

Level 1: No UDP logging occurs.

Level 2: This message records that the source host has sent a UDP datagram to the destination host using the UDP ports specified. The SubID for this signature is the destination UDP port. If the destination host does not have a UDP service at the requested port, an ICMP unreachable message will immediately follow. The default configuration is to not generate this signature unless it is explicitly specified within the configuration file.

Level 3: Same as level 2.

4001 UDP port sweep

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This could be a serious attack. This signature identifies an attacking host that has sent several UDP datagrams to a number of different destination ports on the target host. This method is used by attackers to determine the services available on the target host for potential exploitation.

4100 TFTP Passwd File

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

Packets Required: A large sampling of UDP packets

This signature detects an attempt to access the passwd file via TFTP

6001 Normal SATAN probe

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature is generated when an attacking host has run the tool "SATAN" in normal mode against a target host on the network. Other types of attack activity similar to the method may also cause this signature to be generated.

6002 Heavy SATAN probe

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

This signature is generated when an attacking host has run the tool "SATAN" in heavy mode against a target host on the network. Other types of attack activity similar to the method may also cause this signature to be generated.

6050 DNS HINFO Request

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 2

These signature detects an attempt to access HINFO records from a DNS server. This is commonly used by intruders to determine the types of systems on a network.

6051 DNS Zone Transfer

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 1

These signature detects legitimate DNS zone transfers.

6052 DNS Zone Transfer from High Port

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 4

These signature detects a DNS zone transfer with the source port not equal to 53. This is a common network reconnaissance technique used by intruders.

6053 DNS Request for all Records

SubID Values: 0

Misc Field Info: DNS query name

Recommended Alarm Value: 2

These signature detects a DNS request for all records.

6100 RPC Port Registration

SubID Values: RPC program number

Misc Field Info: none

Recommended Alarm Value: 5

This attack attempts to register new RPC services on a target host.

6101 RPC Port Unregistration

SubID Values: RPC program number

Misc Field Info: none

Recommended Alarm Value: 5

This attack attempts to unregister RPC services on a target host.

6102 RPC Dump

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 4

This is an example of network reconnaissance. This is produced by executing a "rpcinfo -p" against a target host.

6103 Proxied RPC Request

SubID Values: RPC program number

Misc Field Info: none

Recommended Alarm Value: 5

This attack exploits a vulnerability of the portmapper by causing it to forward RPC requests to the local system. This may defeat existing authentication mechanisms. The most widespread use of this vulnerability occurs in tricking mountd on NFS servers to disclose filehandles.

6150 ypserv Attempt

6151 ypbind Attempt

6152 yppasswdd Attempt

6153 ypupdated Attempt

6154 ypxfrd Attempt

6155 mountd Attempt

6175 rexd Attempt

SubID Values: 0

Misc Field Info: none

Recommended Alarm Value: 5

These signatures detect attempts to access the services specified. It is recommended that these services should not be accessed via the Internet.

Regular Expression Based Signatures

Regular expression (regex) based signatures are generated from stateful multi-layer datagram contents analysis. In simple terms - taking at the contents of packets, arranging everything into order, and doing string matching on the results. Any regular expression up to 64 characters in length is supported. This provides an unlimited number of signatures.

While all regex based signatures have a **SigID of 8000,** each string must have a unique SubID assigned to it. Numbering SubIDs is left to the individual installing and maintaining NetRanger. Each string entry also specifies the port number and direction of traffic where it should be searched for. For example, in order to detect attackers using the command "help" on mailservers, the regular expression "[Hh][Ee][Ll][Pp]" should be reported only when it is in the datastream going to port 25 on a target host. This drastically lowers the number of false positives.

Performance is the only limitation to the number of strings that can be simultaneously searched for. Analyzing strings directed at the destination port provides substantially better performance than the reverse. For example, malicious activity inside of telnet sessions can be determined by what an attacker is typing, and not from what appears on the screen. A typical telnet session has over 95% of the network traffic generated going from the port 23 to the user's screen. In this example, better performance is achieved by only analyzing the remaining 5%.

NetSentry Based Signatures

NetSentry based signatures are generated by reporting packet failures and/or successes from NetSentry filters. When the NSC BorderGuard is used to enforce a specific security policy, an option exists to copy a subset of failed packets from the BorderGuard to the NSX. Each copied packet includes the name of the NetSentry filter that failed the packet. NetRanger matches this name with a pre-defined list of filter names and generates a security violation event record upon a match.

```
filter incom1_ip_spoof_fail
   ip_sa mask 0xFF000000 in (10.*.*.*)
      copy_to 10.1.5.3 35399
      fail;
end
```

The above filter was designed to analyze all packets entering a network. If the source address is from the network, it is an impossible packet and may be part of an IP spoofing attack. The filter will send a copy to the NetRanger/NSX at address 10.1.5.3, and then fail the packet. NetRanger will match the string "incom1_ip_spoof_fail" and generate a security violation event record. The event record will include the source and destination IP addresses and applicable ports.

Using this NetRanger capability, any filter on the BorderGuard is capable of generating event signatures. Because the BorderGuard allows a large number of highly configurable filters using the NetSentry filter language, the number of potential NetSentry event signatures generated by NetRanger could be considered unlimited. NetSentry filternames for versions 2.0 thru 3.1 of the BorderGuard OS is case sensitive. Version 4 of the BorderGuard OS and the ERS report filternames in uppercase. If the original filtername is "finger_fail", then NetRanger must be configured to look for the filtername "FILTER_FAIL".

All NetSentry based event records have a SigID of 10000. Each defined filtername in NetRanger must have a unique SubID assigned to it. Numbering SubIDs is left to the individual installing and maintaining NetRanger. The example security filters that come with NetRanger implement the following:

incom1_ip_spoof_fail	IP spoofing attacks
incom1_ip_source_route_fail	IP source routing attacks
incom1_tcp_frag_header_fail	Fragmented TCP header attacks
incom1_tcp_small_frag_fail	Undersized TCP header attacks
incom1_tcp_fail	Generic failed TCP connections
incom1_udp_fail	Generic failed UDP packets
incom1_ftp_fail	Failed FTP attempt
incom1_telnet_fail	Failed telnet attempt
incom1_smtp_fail	Failed e-mail attempt
incom1_dns_fail	Failed Domain Name Server attempt
incom1_gopher_fail	Failed gopher attempt
incom1_finger_fail	Failed finger attempt
incom1_www_fail	Failed WWW attempt
incom1_pop2_fail	Failed POP2 mail attempt
incom1_pop3_fail	Failed POP3 mail attempt
incom1_rpc_fail	Failed RPC attempt
incom1_auth_fail	Failed identd attempt
incom1_nntp_fail	Failed network news attempt
incom1_ntp_fail	Failed network time attempt
incom1_exec_fail	Failed rexec attempt
incom1_login_fail	Failed rlogin attempt
incom1_cmd_fail	Failed rsh attempt
incom1_printer_fail	Failed printer attempt
incom1_ntalk_fail	Failed network talk attempt
incom1_uucp_fail	Failed UUCP attempt
incom1_x11_fail	Failed X11 attempt
incom1_udp_dns_fail	Failed DNS packet
incom1_tftp_fail	Failed TFTP packet
incom1_udp_rpc_fail	Failed RPC packet
incom1_snmp_fail	Failed snmp packet
incom1_syslog_fail	Failed syslog packet

NetRanger Director Configuration and Installation

Pre-Installation for HP-UX systems

Before you begin the installation process, ensure that you meet the Software and Hardware Requirements listed below.

If you are installing the NetRanger/Director on an HP workstation that already has HP-UX 10.10 or greater *and* HP OpenView 4.1 or greater installed, you can skip to the "NetRanger/Director Installation" section.

Software Requirements

You must have the following software either installed on your HP workstation or you must have the following software media and instructions:

- HP-UX 10.10 or greater
- HP OpenView 4.1 or greater

Hardware Requirements

The hardware requirements for the NetRanger/Director are dictated by the Hardware requirements of HP OpenView. Consult the HP OpenView Installation documentation to ensure that your machine is powerful enough to run HP OpenView. In general, it is recommended that you use a dedicated machine that has at least 64 MB of RAM and at least 2 Gig of disk space.

Installing HP-UX 10.10 or greater

Follow the directions in your HP-UX documentation to either install or upgrade to HP-UX 10.10.

Installing HP OpenView 4.1 or greater

HP OpenView will not install correctly if TCP/IP is not functioning properly. The following parameters must be set before you install HP OpenView:

- IP Address
- Hostname
- Subnet mask
- Default gateway hostname
- Default gateway IP address
- system time and timezone

To set the parameters listed above, you can either run the command:

"/etc/set_parms initial"

or follow these steps:

1. **Ensure that the following parameters are set correctly under "General Toolbox.. System_Admin.. SetNetworking":**

From this screen, DNS and NIS can also be configured, but it is best to choose **Cancel** for these options unless you are *certain* you know the proper IP addresses for these services.

CAUTION
If an incorrect IP address is entered for these options, the machine may not reboot properly.

b. **Configure the proper hostname using the following command:**

```
"/etc/set_parms hostname".
```

Do not try to change the hostname by editing /etc/hosts. This will cause the Common Desktop Environment to fail.

c. **Use the "`/etc/set_parms timezone`" command to set the machine's time zone.**

2. **Reboot the machine. Once the machine has rebooted, you should be able to ping your loopback address, ping your IP address, resolve your loopback address, resolve your IP address, and resolve your hostname. Also, the timezone should be correct. Do not go to the next step until TCP/IP is properly configured.**

CAUTION
HP OpenView will not install correctly if TCP/IP is improperly configured.

3. **Install HP OpenView 4.1 or greater on the HP-UX box (see the HP OpenView Installation Manual for details).**
4. **Add the following lines to the /.profile (please note the space between the "." and the "/":**

```
. /opt/OV/bin/ov.envvars.sh
export PATH=$PATH:$OV_BIN
```

Director Installation for HP-UX systems

To install the NetRanger Director software, follow these steps:

1. **Using "`su`", become the root user.**

2. **To load the OpenView environment variables, type the following command:**

```
. /opt/OV/bin/ov.envvars.sh
```

3. **If the OpenView user interface is running, stop it now by selecting Map..Exit from the OpenView menu. If other users have other copies of the user interface running and exported to other displays, ask them to shut down the user interface temporarily.**

4. **Put the NetRanger/Director tape in the tape drive if you have not already done so.**

5. **Go to the /tmp subdirectory by typing:**

```
cd /tmp
```

6. **The NetRanger/Director install tape should contain five compressed .tar files whose names have the following format:**

```
WGCnsx.<version>.<release>.<mod level>.<sys type>.tar.Z
WGCdrctr.<version>.<release>.<mod level>.<sys type>.tar.Z
WGCcfg.<version>.<release>.<mod level>.<sys type>.tar.Z
WGCsapd.<version>.<release>.<mod level>.<sys type>.tar.Z
JDK_<version>_<release>_<mod level>-<sys type>.tar.Z
```

7. **Untar these files using the following syntax (you must run this command for each of the five files):**

```
tar -xvf /dev/rmt/0m <filename>
```

   Where `<filename>` is the name of the compressed tar file.

8. **Uncompress these files using the following syntax (you must run this command for each of the five files):**

```
uncompress <filename>
```

   The files should now be uncompressed and should no longer have the ".Z" extension.

9. **Untar the uncompressed files, *except for the Java Development Kit*, using the following syntax:**

```
tar -xvf <filename>
```

10. **Untar the Java Development Kit using the following commands:**

```
mkdir -p /opt/SUNWjava
cd /opt/SUNWjava
tar -xvf /tmp/JDK_1_0_1-hpux10.tar
mv JDK-1.0.1 JDK
```

11. **Run the install software by typing the following:**

    Installing the NSX Interface software

    ```
    /usr/sbin/swinstall -v -x mount_all_filesystems=false -s /tmp/WGCnsx nsx
    ```

    Installing the RDBMS software

    ```
    /usr/sbin/swinstall -v -x mount_all_filesystems=false -s /tmp/WGCsapd WGCsapd
    ```

    Installing the Network Management Interface software

    ```
    /usr/sbin/swinstall -v -x mount_all_filesystems=false -s /tmp/WGCdrctr
    director
    ```

    Installing the Remote Configuration software

    ```
    /usr/sbin/swinstall -v -x mount_all_filesystems=false -s /tmp/WGCcfg WGCcfg
    ```

12. **Use SAM to set a password for user** netrangr.
13. **If** */usr/nr/tmp* **and** */usr/nr/var* **do not already exist, type the following to create them:**

    ```
    mkdir /usr/nr/tmp
    ```

    ```
    mkdir /usr/nr/var
    ```

14. **Examine the file** */tmp/nrdirmap.install.out* **to ensure that no errors occurred.**

The installation is now complete. Go to the section called "Post-Installation for HP-UX and Sun Solaris Systems."

Pre-Installation for Sun Solaris Systems

Before you begin the installation process, ensure that you meet the Software and Hardware Requirements listed below.

If you are installing the NetRanger/Director on a Sun workstation that already has Solaris 2.4 or greater *and* HP OpenView 4.1 or greater installed, you can go to the "NetRanger/Director Installation" section.

Software Requirements

You must have the following software either installed on your Sun workstation or you must have the following software media and instructions:

- Solaris 2.4 or greater
- HP OpenView 4.1 or greater